BASIC-Interpreter ROM-Listing

```
*
*  BASIC-Interpreter ROM-Listing
*
*  I found this file in the web, thanks to pullmoll
*

*************************************************
* Revision *1.1*, /Fri Jun 18 11:08:09 2004 UTC *
* by /pullmoll/                                 *
*                                               *
*************************************************
-------------------------------------------------------------------------

; ****************************************************************************
; *                    ROM listing Colour BASIC interpreter                  *
; ****************************************************************************
;
; This is a ROM listing of the BASIC interpreter for the EACA EG2000
; Colour Genie computer. There are several versions of the Colour BASIC
; interpreter around. This is the most commonly used.
;
; ---------------------------------------------------------------------------
; Conventions:
;
; A,B,C,D,E,
; H,L,BC,DE,
; HL,IX,IY,
; SP,PC                     Identifiers of the Z80 registers
;
; BCDE                      Concatination of BC and DE registers to hold
;                           a single precision value
;
; X                         Accumulator X (from 411DH onwards)
;
; Y                         Accumulator Y (from 4127H onwards)
;
; Exp ( X )                 Exponent of value in X
;
; Exp ( Y )                 Exponent of value in Y
;
; Exp                       Exponent of any floating point value
;
; Sign                      Sign of a value
;
; VT                        Variable Type
;                           (02 = INT, 03 = STR, 04 = SNG, 08 = DBL)
;
; LP                        Line Pointer (= link)
;
; LN                        Line Number
;
; PTP                       Program Text Pointer
;
; A = B                     A equals B
;
; A -> B                    A points to B (contains address of B)
;
; A,n                       Bit n of register A
;
; I:                        List of entry parameters
;
; O:                        List of exit parameters
;
; SUB                       Subroutine
;
; --                        Unused instruction
;
; '*'                       The following byte is multiple used
;
; -----------------------------------------------------------------------------
; Start of the BASIC interpreter ROMs
```

```
; Entry at 0000H after switch on of the computer or after <RST>+<R>
; Cold start

  00000 F3              DI                      ;Disable interrupts
  00001 AF              XOR     A               ;A = 0
  00002 C37406          JP      0674H           ;Continue at 0674H

  00005 C30040          JP      4000H           ;--

  00008 C30040          JP      4000H           ;RST 08H: jump to 4000H and
                                                ;from there to 1C96H

  0000B E1              POP     HL              ;--
  0000C E9              JP      (HL)

  0000D C30000          JP      0000H           ;--

  00010 C30340          JP      4003H           ;RST 10H: jump to 4003H and
                                                ;from there to 1D78H

; DCB Call for Input (AF,DE)
;
; I: DE = DCB Address
; O: A = Input byte

  00013 C5              PUSH    BC              ;Save BC
  00014 0601            LD      B,01H           ;DCB type: input
  00016 182E            JR      0046H           ;Call DCB


  00018 C30640          JP      4006H           ;RST 18H: jump to 4006H and
                                                ;from there to 1C90H

; DCB Call for Output (AF,DE)
;
; I: DE = DCB Address
; O: A = Output byte

  0001B C5              PUSH    BC              ;Save BC
  0001C 0602            LD      B,02H           ;DCB type: output
  0001E 1826            JR      0046H           ;Call DCB


  00020 C30940          JP      4009H           ;RST 20H: jump to 4009H and
                                                ;from there to 25D9H

; DCB Call for Input/Output (AF,DE) (not used)
;
; I: DE = DCB Address
; O: A = Output byte

  00023 C5              PUSH    BC              ;Save BC
  00024 0604            LD      B,04H           ;DCB type: input/output
  00026 181E            JR      0046H           ;Call DCB


  00028 C30C40          JP      400CH           ;RST 28H: jump to 400CH
                                                ;(BREAK-vector, reserved for
                                                ;DOS)

; Get byte from keyboard (AF,DE)
;
; I: -
; O: A = ASCII code of the key pressed or 00H when no key pressed

  0002B 111540          LD      DE,4015H        ;DE -> Keyboard DCB
  0002E 18E3            JR      0013H           ;Continue at 0013H


  00030 C30F40          JP      400FH           ;RST 38H: jump to 400FH
```

```
                                                  ;(reserved for DOS)

; Output a byte on the screen (AF,DE)
;
; I: A = ASCII code of the character to print on the screen
; O: -

  00033 111D40          LD      DE,401DH        ;DE -> Screen DCB
  00036 18E3            JR      001BH           ;Continue at 001BH


  00038 C31240          JP      4012H           ;RST 38H: jump to 4012H
                                                ;(Interrupt vector, reserved
                                                ;for DOS)

; Output a byte on the printer (AF,DE)
;
; I: A = ASCII code of the character to print on the printer
; O: -

  0003B 112540          LD      DE,4025H        ;DE -> Printer DCB
  0003E 18DB            JR      001BH           ;Continue at 001BH


  00040 C3D905          JP      05D9H           ;Input of a line
                                                ;(see 05D9H)

  00043 C9              RET                     ;--
  00044 00              NOP
  00045 00              NOP

  00046 C3C203          JP      03C2H           ;Jump to DCB call


; Wait for keypress (AF,DE)
;
; I: -
; O: A = ASCII code of key pressed

  00049 CD2B00          CALL    002BH           ;Get byte from keyboard
  0004C B7              OR      A               ;Key pressed ? (A <> 0)
  0004D C0              RET     NZ              ;Yes: return
  0004E 18F9            JR      0049H           ;No: again


; Decoding table for keyboard routine
; ASCII codes of corresponding keys

  00050 0D              DEFB    0DH             ;RETURN
  00051 0D              DEFB    0DH             ;RETURN SHIFT
  00052 1F              DEFB    1FH             ;CLEAR
  00053 1F              DEFB    1FH             ;CLEAR SHIFT
  00054 01              DEFB    01H             ;BREAK
  00055 01              DEFB    01H             ;BREAK SHIFT
  00056 5B              DEFB    5BH             ;UP ARROW
  00057 1B              DEFB    1BH             ;UP ARROW SHIFT
  00058 0A              DEFB    0AH             ;DOWN ARROW
  00059 1A              DEFB    1AH             ;DOWN ARROW SHIFT
  0005A 08              DEFB    08H             ;LEFT ARROW
  0005B 18              DEFB    18H             ;LEFT ARROW SHIFT
  0005C 09              DEFB    09H             ;RIGHT ARROW
  0005D 19              DEFB    19H             ;RIGHT ARROW SHIFT
  0005E 20              DEFB    20H             ;SPACE BAR
  0005F 20              DEFB    20H             ;SPACE BAR SHIFT


; Delay loop (AF,BC)
;
; I: BC = counter (delay time = BC * 11.3 microseconds)
; O: -
```

```
  00060 0B                DEC     BC              ;Decrement counter
  00061 78                LD      A,B
  00062 B1                OR      C               ;Counter zero ?
  00063 20FB              JR      NZ,0060H        ;No: loop
  00065 C9                RET


; RESET entry after pressing both <RST> keys

  00066 01181A            LD      BC,1A18H        ;BC -> entry back into BASIC
  00069 C3CA05            JP      05CAH           ;Continue at 05CAH


; Start 4 (continuation of 05C7H)
; Prepare RAM for BASIC

  0006C 31F841            LD      SP,41F8H        ;Set stackpointer
  0006F 118040            LD      DE,4080H        ;Copy ROM area of 18F7H
  00072 21F718            LD      HL,18F7H        ;to 191CH into RAM area
  00075 012700            LD      BC,0027H        ;from 4080H to 40A6H
  00078 C34001            JP      0140H           ;Continue at 0140H


; Start 6 when no ROM cartridge with a BASIC program is present
; (continuation of 0149H)

  0007B 210158            LD      HL,5801H        ;HL -> start of BASIC program
  0007E 3A80F8            LD      A,(0F880H)
  00081 CB4F              BIT     1,A             ;<MOD SEL> pressed ?
  00083 2009              JR      NZ,008EH        ;Yes: continue at 008EH

  00085 22A440            LD      (40A4H),HL      ;Save pointer in system RAM
  00088 CD4638            CALL    3846H           ;FCLS


; Start 6 when a ROM cartridge with a BASIC program is present
; (continuation of 014CH)

  0008B 22A440            LD      (40A4H),HL      ;Save pointer to start of
                                                  ;program in system RAM
  0008E 21E541            LD      HL,41E5H        ;HL -> (start of line buffer-3)
                                                  ;Mark start:
  00091 363A              LD      (HL),3AH        ;':'
  00093 23                INC     HL
  00094 70                LD      (HL),B          ;00H
  00095 23                INC     HL
  00096 362C              LD      (HL),2CH        ;'.'
  00098 23                INC     HL
  00099 22A740            LD      (40A7H),HL      ;Save line buffer pointer in
                                                  ;system RAM

  0009C 113B01            LD      DE,013BH        ;DE -> error routine

                                                  ;Block Disk BASIC vectors
  0009F 061C              LD      B,1CH           ;28 vectors
  000A1 215241            LD      HL,4152H        ;HL -> vector table in RAM
                                                  ;Construct vectors:
                                                  ;28 times a JP 013BH in RAM
  000A4 36C3              LD      (HL),0C3H       ;JP opcode
  000A6 23                INC     HL
  000A7 73                LD      (HL),E          ;LSB of error routine address
  000A8 23                INC     HL
  000A9 72                LD      (HL),D          ;MSB of error routine address
  000AA 23                INC     HL
  000AB 10F7              DJNZ    00A4H           ;Next vector

                                                  ;Remaining DOS vectors are
                                                  ;blocked with a RET
  000AD 0615              LD      B,15H           ;21 vectors
  000AF 36C9              LD      (HL),0C9H       ;Insert RET opcode
  000B1 23                INC     HL              ;Reserve 2 bytes RAM space
```

```
  000B2 23              INC     HL              ;so that the RET can be changed
  000B3 23              INC     HL              ;to a JP instruction
  000B4 10F9            DJNZ    00AFH           ;Next vector

  000B6 2AA440          LD      HL,(40A4H)      ;HL -> start of BASIC program
  000B9 2B              DEC     HL              ;Adjust pointer
  000BA 70              LD      (HL),B          ;Mark start - 1 with 00H
  000BB CD7038          CALL    3870H           ;Initialize CRTC
  000BE CD8F1B          CALL    1B8FH           ;Set stack pointer
  000C1 CDAF06          CALL    06AFH           ;Test ROM cartridge present
  000C4 CDC901          CALL    01C9H           ;CLS
  000C7 211801          LD      HL,0118H        ;HL -> 'MEM SIZE'
  000CA CDA728          CALL    28A7H           ;Print text
  000CD CDB31B          CALL    1BB3H           ;Print '?' and wait for
                                                ;input
                                                ;<BREAK> pressed ?
  000D0 38F5            JR      C,00C7H         ;Yes: repeat input
  000D2 D7              RST     10H             ;Get next non-space character
  000D3 B7              OR      A               ;Only <RETURN> entered ?
  000D4 2013            JR      NZ,00E9H        ;No: an number was entered,
                                                ;continue a 00E9H
  000D6 210040          LD      HL,4000H        ;Yes: HL -> start of RAM
  000D9 23              INC     HL              ;HL + 1
  000DA 7C              LD      A,H             ;HL = 0000 ?
  000DB B5              OR      L               ;(Complete memory tested)
  000DC 281C            JR      Z,00FAH         ;Yes: continue at 00FAH
  000DE 7E              LD      A,(HL)          ;Take byte from memory
  000DF 47              LD      B,A             ;Save it in B
  000E0 2F              CPL                     ;Complement A
  000E1 77              LD      (HL),A          ;Put it back in memory
  000E2 BE              CP      (HL)            ;Is it really in RAM ?
  000E3 70              LD      (HL),B          ;Restore original value
  000E4 28F3            JR      Z,00D9H         ;Yes: next byte
                                                ;No: HL -> end of RAM + 1
  000E6 25              DEC     H               ;Subtract 256 bytes for
                                                ;SHAPE table
  000E7 1811            JR      00FAH           ;Continue at 00FAH

; Enter number at MEM SIZE?

  000E9 CD5A1E          CALL    1E5AH           ;Evaluate entered number
  000EC B7              OR      A               ;Only digits entered ?
  000ED C29719          JP      NZ,1997H        ;No: ?SN Error

  000F0 EB              EX      DE,HL
  000F1 2B              DEC     HL              ;HL = MEM SIZE - 1
  000F2 3E8F            LD      A,8FH           ;A = test byte
  000F4 46              LD      B,(HL)          ;Save original byte
  000F5 77              LD      (HL),A          ;Testbyte to (HL)
  000F6 BE              CP      (HL)            ;Is it really saved ?
  000F7 70              LD      (HL),B          ;Put original value back
  000F8 20CD            JR      NZ,00C7H        ;No: again!

; Store memory boundaries
; HL -> highest available memory location - 1

  000FA 2B              DEC     HL              ;HL - 1
  000FB 111444          LD      DE,4414H        ;DE = 4414H
  000FE DF              RST     18H             ;Compare HL with DE
                                                ;MEM SIZE < 4414H ?
  000FF DA7A19          JP      C,197AH         ;Yes: ?OM Error
  00102 11CEFF          LD      DE,0FFCEH       ;DE = -50
  00105 22B140          LD      (40B1H),HL      ;Store TOPMEM
  00108 19              ADD     HL,DE           ;HL = HL - 50: CLEAR 50
  00109 22A040          LD      (40A0H),HL      ;Save start address of
                                                ;string space
  0010C CD4D1B          CALL    1B4DH           ;NEW and CLEAR
  0010F 212101          LD      HL,0121H        ;HL -> 'COLOUR BASIC'
  00112 CDA728          CALL    28A7H           ;Print text
  00115 C3191A          JP      1A19H           ;Jump to BASIC active command
                                                ;mode
```

```
  0015D CD8301           CALL    0183H          ;Get bitnr. and address
  00160 3EFE             LD      A,0FEH         ;A = bit mask
  00162 07               RLCA                   ;Shift A until specified bit
  00163 10FD             DJNZ    0162H          ;is reset in A
  00165 0F               RRCA                   ;Ajust because B was bitnr + 1
  00166 47               LD      B,A            ;B = bit mask
  00167 1A               LD      A,(DE)         ;Take byte from memory
  00168 A0               AND     B              ;Reset bit
  00169 12               LD      (DE),A         ;Put byte back in memory
  0016A C9               RET


; Continuation of CHECK routine of 0132H

  0016B D7               RST     10H             ;Get next non-space caharacter
  0016C CF               RST     08H             ;Next byte must be
  0016D 28               DEFB    '('             ;a '('
  0016E CD8301           CALL    0183H           ;Get bitnr and address
  00171 E5               PUSH    HL              ;Save PTP
  00172 1A               LD      A,(DE)          ;Get byte from memory
  00173 1F               RRA                     ;and specified bit in C-flag
  00174 10FD             DJNZ    0173H           ;Loop
  00176 21FFFF           LD      HL,0FFFFH       ;HL = -1 (TRUE)
                                                 ;Bit set ?
  00179 3801             JR      C,017CH         ;Yes: continue at 017CH
  0017B 23               INC     HL              ;HL = 0 (FALSE)
  0017C CD9A0A           CALL    0A9AH           ;Write HL to X as INT
  0017F E1               POP     HL              ;Restore PTP
  00180 CF               RST     08H             ;Next byte must be
  00181 29               DEFB    ')'             ;')'
  00182 C9               RET


; SUB for SET, RESET and CHECK
; Get bit number and address form program text
;
; I: HL = PTP
; O: B = Bit number + 1
;    DE = Address

  00183 CD1C2B           CALL    2B1CH           ;A = Bit number
  00186 FE08             CP      08H             ;Out of range ?
  00188 D24A1E           JP      NC,1E4AH        ;Yes: ?FC Error
  0018B F5               PUSH    AF              ;Save bit number
  0018C CF               RST     08H             ;Both parameters separated by
  0018D 2C               DEFB    ','             ;a comma ?
  0018E CD022B           CALL    2B02H           ;DE = address
  00191 F1               POP     AF              ;Restore bit number
  00192 47               LD      B,A             ;Put it in B
  00193 04               INC     B               ;B = bit number + 1
  00194 C9               RET


; Unused ROM space

  00195 FF               RST     38H             ;--
  00196 FF               RST     38H
  00197 FF               RST     38H
  00198 FF               RST     38H
  00199 FF               RST     38H
  0019A FF               RST     38H
  0019B FF               RST     38H
  0019C FF               RST     38H


; X = INKEY$
; ----------

  0019D D7               RST     10H             ;Get next non-space character
  0019E E5               PUSH    HL              ;Save PTP
  0019F 3A9940           LD      A,(4099H)       ;Has a key already been pressed
```

```
  001A2 B7                OR      A                ;before ?
  001A3 2006              JR      NZ,01ABH         ;Yes, pass value and return

  001A5 CD5803            CALL    0358H            ;No: key pressed now ?
  001A8 B7                OR      A
  001A9 2811              JR      Z,01BCH          ;No: null string as result

  001AB F5                PUSH    AF               ;Save key code
  001AC AF                XOR     A                ;Delete last key code
  001AD 329940            LD      (4099H),A
  001B0 3C                INC     A                ;Reserve a byte in
  001B1 CD5728            CALL    2857H            ;string space
  001B4 F1                POP     AF               ;A = key code
  001B5 2AD440            LD      HL,(40D4H)       ;HL -> memory for new string
  001B8 77                LD      (HL),A           ;Save string
  001B9 C38428            JP      2884H            ;Set VT to STR

; No key pressed
; Return a null string as result

  001BC 212819            LD      HL,1928H         ;HL -> null string
  001BF 222141            LD      (4121H),HL       ;Set as result
  001C2 3E03              LD      A,03H            ;Set VT to STR
  001C4 32AF40            LD      (40AFH),A
  001C7 E1                POP     HL               ;Restore PTP
  001C8 C9                RET


; CLS statement
; -------------

  001C9 3E1C              LD      A,1CH            ;A = cursor home
  001CB CD3A03            CALL    033AH            ;Output A
  001CE 3E1F              LD      A,1FH            ;A = clear screen from
                                                   ;cursor pos. to end
  001D0 C33A03            JP      033AH            :Output A


; RANDOM statement
; ----------------

  001D3 ED5F              LD      A,R              ;A = refresh register
  001D5 32AB40            LD      (40ABH),A        ;Put it in system RAM
  001D8 C9                RET


; Issue a level change to cassette port

  001D9 3A1C43            LD      A,(431CH)        ;A = last written value
  001DC EE01              XOR     01H              ;Toggle bit 0
  001DE D3FF              OUT     (0FFH),A         ;Output value to cassette port
  001E0 321C43            LD      (431CH),A        ;and save it is system RAM
  001E3 C9                RET

; blink '*'

  001E4 3A2744            LD      A,(4427H)        ;Get '*' or ' ' from screen
  001E7 EE0A              XOR     0AH              ;Swap '*' and ' '
  001E9 322744            LD      (4427H),A        ;Put it back on screen
  001EC C9                RET

; Read one byte from cassette (AF)
;
; I: -
; O: A = byte read from cassette port

  001ED D9                EXX                      ;Exchange registers

  001EE 0608              LD      B,08H            ;Read 8 bits
  001F0 1600              LD      D,00H            ;Put byte to 0
  001F2 CDFA01            CALL    01FAH            ;Read bit and shift into D
```

```
  001F5 10FB              DJNZ    01F2H             ;Next bit

  001F7 7A                LD      A,D               ;A = complete byte read
  001F8 D9                EXX                       ;Exchange registers
  001F9 C9                RET


; Read one bit from cassette and shift into D

  001FA C5                PUSH    BC                ;Save counter
                                                    ;Wait for clock pulse:
  001FB DBFF              IN      A,(0FFH)          ;Test port FFH
  001FD E601              AND     01H               ;Mask level
  001FF 5F                LD      E,A               ;and save in E

  00200 DBFF              IN      A,(0FFH)          ;Test port FFH
  00202 E601              AND     01H               ;Mask level
  00204 AB                XOR     E                 ;Compare with previous level
  00205 1F                RRA                       ;Put result in C-flag
  00206 30F8              JR      NC,0200H          ;Search clock pulse

  00208 3C                INC     A                 ;A = 1
  00209 AB                XOR     E                 ;A = new level value
  0020A 5F                LD      E,A               ;Store it in E for compare
  0020B 3A1243            LD      A,(4312H)         ;A = delay time
  0020E 47                LD      B,A               ;Put it in B
  0020F 10FE              DJNZ    020FH             ;Delay loop

  00211 DBFF              IN      A,(0FFH)          ;Now take new input level
  00213 E601              AND     01H
  00215 AB                XOR     E                 ;and compare it with previous
                                                    ;level. A = 1 if a level change
                                                    ;is recognized.
  00216 CB22              SLA     D                 ;Shift D
  00218 B2                OR      D                 ;and add new value
  00219 57                LD      D,A               ;Result in D
  0021A C1                POP     BC                ;Restore counter
  0021B C9                RET


; Write the same byte twice to cassette

  0021C CD1F02            CALL    021FH             ;Write first byte


; Write one byte to cassette (AF)
;
; I: A = byte to be written
; O: A = byte written

  0021F D9                EXX                       ;Save registers
  00220 F5                PUSH    AF                ;Save byte
  00221 0E08              LD      C,08H             ;8 bits to write
  00223 57                LD      D,A               ;D = byte
  00224 CDD901            CALL    01D9H             ;Issue clock pulse
  00227 3A1043            LD      A,(4310H)         ;A = delay value
  0022A 47                LD      B,A               ;B = A
  0022B 10FE              DJNZ    022BH             ;Delay loop

  0022D 7A                LD      A,D               ;A = byte
  0022E 07                RLCA                      ;Shift next bit into C-Flag
  0022F 57                LD      D,A               ;And save shifted value in D
  00230 DCD901            CALL    C,01D9H           ;Bit = 1: Issue level change
  00233 3A1143            LD      A,(4311H)         ;A = delay value
  00236 47                LD      B,A               ;B = A
  00237 10FE              DJNZ    0237H             ;Delay loop

  00239 0D                DEC     C                 ;Decrement bit counter
                                                    ;All bits done ?
  0023A 20E8              JR      NZ,0224H          ;No: loop, issue next bit
  0023C F1                POP     AF                ;Restore byte value
```

```
  0023D D9              EXX                     ;Restore registers
  0023E C9              RET


; Write leader and sync on cassette

  0023F 06FF            LD      B,0FFH          ;write 255 times
  00241 3EAA            LD      A,0AAH          ;the byte AAH
  00243 CD1F02          CALL    021FH           ;on cassette tape
  00246 10FB            DJNZ    0243H

  00248 3E66            LD      A,66H           ;Sync = 66H
  0024A 18D3            JR      021FH           ;Write on cassette tape


; Search for leader and sync

  0024C E5              PUSH    HL              ;Save registers
  0024D D5              PUSH    DE
  0024E C5              PUSH    BC

  0024F 216935          LD      HL,3569H        ;HL -> colour code table
  00252 110000          LD      DE,0000H        ;DE = 0000H
  00255 3A2340          LD      A,(4023H)       ;A = present colour value
  00258 5F              LD      E,A             ;DE = offset in table
  00259 19              ADD     HL,DE           ;HL -> colour code
  0025A 7E              LD      A,(HL)          ;A = colour code from table
  0025B 3226F0          LD      (0F026H),A      ;Store in colour RAM at the
  0025E 3227F0          LD      (0F027H),A      ;location for both asterisks

  00261 01AA80          LD      BC,80AAH        ;B = counter = 128
                                                ;C = byte to search for
  00264 CDFA01          CALL    01FAH           ;Read bit from cassette,
                                                ;shift into D and load in A
  00267 B9              CP      C               ;Byte found ?
  00268 20F7            JR      NZ,0261H        ;No: search again

  0026A 3EFF            LD      A,0FFH          ;Yes: invert all bits in C
  0026C A9              XOR     C               ;so search for 55H now
  0026D 4F              LD      C,A             ;Result in C
  0026E 10F4            DJNZ    0264H           ;Continue search until 128
                                                ;times AAH found
  00270 CDFA01          CALL    01FAH           ;Search sync (66H)
  00273 FE66            CP      66H             ;Found ?
  00275 20F9            JR      NZ,0270H        ;No: continue seach

  00277 3E2A            LD      A,2AH           ;A = '*'
  00279 322644          LD      (4426H),A       ;Put both asterisks on screen
  0027C 322744          LD      (4427H),A

  0027F C1              POP     BC              ;Restore registers
  00280 D1              POP     DE
  00281 E1              POP     HL
  00282 C9              RET


; Unused ROM area

  00283 FF              RST     38H             ;--
  00284 FF              RST     38H
  00285 FF              RST     38H
  00286 FF              RST     38H
  00287 FF              RST     38H
  00288 FF              RST     38H
  00289 FF              RST     38H
  0028A FF              RST     38H
  0028B FF              RST     38H
  0028C FF              RST     38H
  0028D FF              RST     38H
  0028E FF              RST     38H
  0028F FF              RST     38H
```

```
  00290 FF              RST     38H
  00291 FF              RST     38H
  00292 FF              RST     38H
  00293 FF              RST     38H
  00294 FF              RST     38H
  00295 FF              RST     38H
  00296 FF              RST     38H
  00297 FF              RST     38H
  00298 FF              RST     38H
  00299 FF              RST     38H
  0029A FF              RST     38H
  0029B FF              RST     38H
  0029C FF              RST     38H
  0029D FF              RST     38H
  0029E FF              RST     38H
  0029F FF              RST     38H
  002A0 FF              RST     38H
  002A1 FF              RST     38H
  002A2 FF              RST     38H
  002A3 FF              RST     38H
  002A4 FF              RST     38H
  002A5 FF              RST     38H
  002A6 FF              RST     38H
  002A7 FF              RST     38H
  002A8 FF              RST     38H
  002A9 FF              RST     38H
  002AA FF              RST     38H
  002AB FF              RST     38H


; Read start address of loaded program from tape (for SYSTEM)

  002AC CD1403          CALL    0314H           ;Read 2 bytes (start address)
  002AF 22DF40          LD      (40DFH),HL      ;and store it in system RAM


; SYSTEM statement
; ----------------

  002B2 CDE241          CALL    41E2H           ;DOS
  002B5 318842          LD      SP,4288H        ;Put stack in keyboard buffer
                                                ;to get it out of the way
  002B8 CDFE20          CALL    20FEH           ;Start new line
  002BB 3E2A            LD      A,2AH           ;A = '*'
  002BD CD2A03          CALL    032AH           ;Print it
  002C0 CDB31B          CALL    1BB3H           ;Ask for filename
  002C3 DA6600          JP      C,0066H         ;Back to BASIC
                                                ;if <BREAK> key pressed

  002C6 D7              RST     10H             ;Get next non-space character
                                                ;Anything entered ?
  002C7 CA9719          JP      Z,1997H         ;No: ?SN Error
  002CA FE2F            CP      '/'             ;Is it a '/' ?
  002CC 284F            JR      Z,031DH         ;Yes: continue at 031DH

  002CE CD4C02          CALL    024CH           ;No: search for leader and sync
  002D1 CDED01          CALL    01EDH           ;Get first byte from tape
  002D4 FE55            CP      55H             ;Is it a SYSTEM program ?
  002D6 20F9            JR      NZ,02D1H        ;No: continue search

  002D8 0606            LD      B,06H           ;6 characters for filename

  002DA 7E              LD      A,(HL)          ;Get character from buffer
  002DB B7              OR      A               ;End of line ?
  002DC 2809            JR      Z,02E7H         ;Yes: load program

  002DE CDED01          CALL    01EDH           ;No: load filename character
  002E1 BE              CP      (HL)            ;Same as in specified name ?
  002E2 20ED            JR      NZ,02D1H        ;No: search next file

  002E4 23              INC     HL              ;Yes: pointer + 1
```

```
  002E5 10F3            DJNZ    02DAH           ;Check next filename char.

  002E7 CDE401          CALL    01E4H           ;Blink right asterisk
  002EA CDED01          CALL    01EDH           ;Read block marker from tape
  002ED FE78            CP      78H             ;End found ?
  002EF 28BB            JR      Z,02ACH         ;Yes: get start address and
                                                ;issue '*?' again
  002F1 FE3C            CP      3CH             ;Start of block ?
  002F3 20F5            JR      NZ,02EAH        ;No: search new byte

  002F5 CDED01          CALL    01EDH           ;Read block length
  002F8 47              LD      B,A             ;B = block length
  002F9 CD1403          CALL    0314H           ;HL = block address
  002FC 85              ADD     A,L             ;Calculate checksum
  002FD 4F              LD      C,A             ;Store in C
  002FE CDED01          CALL    01EDH           ;Read byte
  00301 77              LD      (HL),A          ;Store it in memory
  00302 23              INC     HL              ;Memory pointer + 1
  00303 81              ADD     A,C             ;Add to checksum
  00304 4F              LD      C,A             ;Store result in C
  00305 10F7            DJNZ    02FEH           ;Load next byte from block

  00307 CDED01          CALL    01EDH           ;Block is done: get checksum
  0030A B9              CP      C               ;Same as calculated checksum ?
  0030B 28DA            JR      Z,02E7H         ;Yes: load next block

  0030D 3E43            LD      A,43H           ;No: A = 'C'
  0030F 322644          LD      (4426H),A       ;Indicate checksum error
  00312 18D6            JR      02EAH           ;Continue loading anyway

; SUB for SYSTEM
; Load address (block address or start address) from tape
;
; I: -
; O: HL = address read from tape

  00314 CDED01          CALL    01EDH           ;Load LSB from tape
  00317 6F              LD      L,A             ;L = LSB of address
  00318 CDED01          CALL    01EDH           ;Load MSB from tape
  0031B 67              LD      H,A             ;H = MSB of address
  0031C C9              RET

; Issue '/' with SYSTEM

  0031D EB              EX      DE,HL           ;DE -> text
  0031E 2ADF40          LD      HL,(40DFH)      ;HL = start address
  00321 EB              EX      DE,HL           ;Load DE
  00322 D7              RST     10H             ;Get next non-space character
                                                ;Start address entered ?
  00323 C45A1E          CALL    NZ,1E5AH        ;Yes: new value into DE
  00326 208A            JR      NZ,02B2H        ;Restart SYSTEM if rubbish
                                                ;was entered

  00328 EB              EX      DE,HL           ;Load address into HL
  00329 E9              JP      (HL)            ;Execute program


; Output of A on screen, printer or cassette (AF,HL)
;
; I: A = ASCII value of the byte to be output
;    (409CH) = output flag: 00H = screen, 01H = printer, 80H = cassette

  0032A C5              PUSH    BC              ;Save BC
  0032B 4F              LD      C,A             ;C = output byte

  0032C CDC141          CALL    41C1H           ;DOS
  0032F 3A9C40          LD      A,(409CH)       ;Test flag
  00332 B7              OR      A
  00333 79              LD      A,C             ;Byte back in A
  00334 C1              POP     BC              ;Restore  BC
  00335 C36405          JP      0564H           ;Continue at 0564H
```

```
; Unused ROM space

  00338 FF               RST     38H              ;--
  00339 FF               RST     38H


; Output A on screen and increment POS (40A6H)

  0033A D9               EXX                      ;Save registers
  0033B F5               PUSH    AF               ;Save byte
  0033C CD3300           CALL    0033H            ;Output byte
  0033F CD4803           CALL    0348H            ;Calculate new POS value
  00342 32A640           LD      (40A6H),A        ;and save it in system RAM
  00345 F1               POP     AF               ;Restore byte
  00346 D9               EXX                      ;Restore registers
  00347 C9               RET

; Calculate new POS (AF,DE)
; I: -
; O: A = new POS value

  00348 E5               PUSH    HL               ;Save PTP
  00349 2A2040           LD      HL,(4020H)       ;HL = cursor address
  0034C 110044           LD      DE,4400H         ;DE -> start of screen memory
  0034F B7               OR      A                ;C-flag = 0
  00350 C3D904           JP      04D9H            ;Continue at 04D9H


; Unused ROM space

  00353 FF               RST     38H              ;--
  00354 FF               RST     38H


; Calculate new POS (as 0348H)

  00355 C39D30           JP      309DH            ;continue at 309DH


; Keyboard scan (AF)
; (as 002B but with DOS and saving DE)

  00358 CDC441           CALL    41C4H            ;DOS
  0035B D5               PUSH    DE               ;Save DE
  0035C CD2B00           CALL    002BH            ;Scan keyboard
  0035F D1               POP     DE               ;Restore DE
  00360 C9               RET


; Input of a line with max. 240 characters in line buffer (AF,DE,HL)
;
; I: -
; O: C-flag = 1: <BREAK> key pressed.
;    HL -> start of line buffer - 1

  00361 AF               XOR     A                ;A = 00H
  00362 329940           LD      (4099H),A        ;Delete last key code
  00365 32A640           LD      (40A6H),A        ;POS = 0
  00368 CDAF41           CALL    41AFH            ;DOS
  0036B C5               PUSH    BC               ;Save BC
  0036C 2AA740           LD      HL,(40A7H)       ;HL -> start of line buffer
  0036F 06F0             LD      B,0F0H           ;B = max. number of characters
  00371 CDD905           CALL    05D9H            ;Enter line
  00374 F5               PUSH    AF               ;Save flags
  00375 48               LD      C,B              ;C = number of entered chars.
  00376 0600             LD      B,00H            ;BC = length of line
  00378 09               ADD     HL,BC            ;HL -> last character + 1
  00379 3600             LD      (HL),00H         ;Terminate line with 00H
  0037B 2AA740           LD      HL,(40A7H)       ;HL -> start of line buffer
```

```
  0037E F1              POP     AF              ;Restore flags
  0037F C1              POP     BC              ;Restore BC
  00380 2B              DEC     HL              ;Adjust HL for RST 10H
                                                ;<BREAK> pressed ?
  00381 D8              RET     C               ;Yes: return

  00382 AF              XOR     A               ;C-flag = 0
  00383 C9              RET


; Wait for key pressed (AF)
; (as 0049H but with DOS and save DE)

  00384 CD5803          CALL    0358H           ;Get key
  00387 B7              OR      A               ;New key pressed ?
  00388 C0              RET     NZ              ;Yes: return
  00389 18F9            JR      0384H           ;No: wait for key pressed


; End output to printer (AF)
;
; I: -
; O: PPOS = 0 (Printer POS)

  0038B AF              XOR     A               ;Set output flag
  0038C 329C40          LD      (409CH),A       ;to screen output
  0038F 3A9B40          LD      A,(409BH)       ;A = PPOS
  00392 B7              OR      A               ;Zero ?
  00393 C8              RET     Z               ;Yes: return

  00394 3E0D            LD      A,0DH           ;No: CR to printer
  00396 D5              PUSH    DE              ;Save DE
  00397 CD9C03          CALL    039CH           ;Output CR
  0039A D1              POP     DE              ;Restore DE
  0039B C9              RET


; Print one character on printer and  intercept control characters

  0039C F5              PUSH    AF              ;Save registers
  0039D D5              PUSH    DE
  0039E C5              PUSH    BC
  0039F 4F              LD      C,A             ;Save character in C
  003A0 1E00            LD      E,00H           ;E = 00H for new PPOS
  003A2 FE0C            CP      0CH             ;Is it a Form Feed ?
  003A4 2810            JR      Z,03B6H         ;Yes: PPOS to 0 and print
                                                ;character
  003A6 FE0A            CP      0AH             ;Is it a Line Feed ?
  003A8 2003            JR      NZ,03ADH        ;No: continue test

  003AA 3E0D            LD      A,0DH           ;Convert LF into CR
  003AC 4F              LD      C,A             ;Store it in C

  003AD FE0D            CP      0DH             ;Is it a Carriage Return ?
  003AF 2805            JR      Z,03B6H         ;Yes: PPOS to 0 and print
                                                ;character
  003B1 3A9B40          LD      A,(409BH)       ;No: get PPOS
  003B4 3C              INC     A               ;and increment
  003B5 5F              LD      E,A             ;Store in E
  003B6 7B              LD      A,E             ;Save new PPOS
  003B7 329B40          LD      (409BH),A       ;in system RAM
  003BA 79              LD      A,C             ;A = character to print
  003BB CD3B00          CALL    003BH           ;Print it
  003BE C1              POP     BC              ;Restore registers
  003BF D1              POP     DE
  003C0 F1              POP     AF
  003C1 C9              RET


; DCB call (AF) (see 0046H)
;
```

```
; I: A = character (when output)
;    B = DCB type
;    DE -> DCB
; O: A = character (when input)

  003C2 E5              PUSH    HL              ;Save registers
  003C3 DDE5            PUSH    IX
  003C5 D5              PUSH    DE
  003C6 DDE1            POP     IX              ;IX -> DCB
  003C8 D5              PUSH    DE              ;Adjust stack
  003C9 21DD03          LD      HL,03DDH        ;HL = new return address
  003CC E5              PUSH    HL              ;Put it in stack
  003CD 4F              LD      C,A             ;Save character in C
  003CE 1A              LD      A,(DE)          ;Load DCB type
  003CF A0              AND     B               ;Mask unused bits
  003D0 B8              CP      B               ;Same as indicated type ?
  003D1 C23340          JP      NZ,4033H        ;No: false DCB type, set A
                                                ;to 00H and back to program
                                                ;(4033H is used by DOS)
  003D4 FE02            CP      02H             ;DCB type = output ?
  003D6 DD6E01          LD      L,(IX01H)       ;HL = address of DCB routine
  003D9 DD6602          LD      H,(IX02H)
  003DC E9              JP      (HL)            ;Execute DCB routine

; End of DCB routine

  003DD D1              POP     DE              ;Restore registers
  003DE DDE1            POP     IX
  003E0 E1              POP     HL
  003E1 C1              POP     BC
  003E2 C9              RET


; Keyboard routine (called by DCB)

  003E3 CDAF06          CALL    06AFH           ;ROM catrigde present ?
                                                ;Yes: start ROM program
  003E6 3A80F8          LD      A,(0F880H)      ;Scan keyboard
  003E9 FE12            CP      12H             ;<CTRL> and <MOD SEL> pressed ?
  003EB 200D            JR      NZ,03FAH        ;No: continue at 03FAH

  003ED CDA938          CALL    38A9H           ;FGR

  003F0 3A40F8          LD      A,(0F840H)      ;Scan keyboard
  003F3 CB57            BIT     2,A             ;<BREAK> pressed ?
  003F5 28F9            JR      Z,03F0H         ;No: wait for <BREAK>

  003F7 CDB038          CALL    38B0H           ;LGR
  003FA 213640          LD      HL,4036H        ;HL -> temporary memory
  003FD 0101F8          LD      BC,0F801H       ;BC = first keyboard address
  00400 1600            LD      D,00H           ;Line counter = 0
  00402 0A              LD      A,(BC)          ;Scan keyboard
  00403 5F              LD      E,A             ;Result in E
  00404 AE              XOR     (HL)            ;Set only the bits for those
                                                ;keys, that have changed since
                                                ;last scan
  00405 73              LD      (HL),E          ;Store new code for next scan
  00406 A3              AND     E               ;Set only the bit for the key
                                                ;key that was newly pressed
                                                ;since last scan
  00407 2010            JR      NZ,0419H        ;Was a key pressed ?
                                                ;Yes: find ASCII code of key
  00409 14              INC     D               ;No: line counter + 1
  0040A 2C              INC     L               ;Buffer pointer  + 1
  0040B CB01            RLC     C               ;Update keyboard address
  0040D 30F3            JR      NC,0402H        ;Scan next keyboard address

  0040F 3A80F8          LD      A,(0F880H)      ;Scan keyboard
  00412 CB5F            BIT     3,A             ;<RPT> pressed ?
  00414 C2D404          JP      NZ,04D4H        ;Yes: continue at 04D4H
```

```
  00417 AF                XOR     A               ;A = 00H
  00418 C9                RET

; Find ASCII code of key

  00419 5F                LD      E,A             ;Bit value to E
  0041A 211840            LD      HL,4018H        ;HL = flag byte
  0041D 3A80F8            LD      A,(0F880H)      ;Scan keyboard
  00420 CB4F              BIT     1,A             ;<MOD SEL> pressed ?
  00422 C2C904            JP      NZ,04C9H        ;Yes: continue at 04C9H

  00425 CB67              BIT     4,A             ;<CTRL> pressed ?
  00427 C2D004            JP      NZ,04D0H        ;Yes: continue at 04D0H

  0042A 3E07              LD      A,07H           ;Was the key in the last
                                                  ;keyboard address and was not
                                                  ;intercepted ?
  0042C BA                CP      D               ;Line counter = 7 ?
  0042D 28E8              JR      Z,0417H         ;Yes: the key pressed was
                                                  ;the <SHIFT> key (code = 00H)
  0042F 7A                LD      A,D             ;Multiply D by 8
  00430 07                RLCA                    ;* 2
  00431 07                RLCA                    ;* 2
  00432 07                RLCA                    ;* 2
  00433 57                LD      D,A             ;Because there are 8 keys per
                                                  ;keyboard address
  00434 0E01              LD      C,01H           ;C = mask for calculation
  00436 79                LD      A,C             ;A = bit mask
  00437 A3                AND     E               ;Mask key code
                                                  ;Key newly pressed ?
  00438 2005              JR      NZ,043FH        ;Yes: continue at 043FH

  0043A 14                INC     D               ;Increment D to establish value
                                                  ;of key in keyboard matrix
  0043B CB01              RLC     C               ;Shift mask for next test
  0043D 18F7              JR      0436H

; D now indicates the value of the key in the keyboard matrix
; @ has value 00H, A has value 01H etc., upto the space bar which has
; a value of 37H

  0043F 3A80F8            LD      A,(0F880H)      ;Scan keyboard
  00442 47                LD      B,A             ;Save value in B
  00443 7A                LD      A,D             ;A = matrix value
  00444 C640              ADD     A,40H           ;Matrix value + 40H gives
                                                  ;correct ASCII code for the
                                                  ;kes '@' to 'F4' (ASCII codes
                                                  ;40H to 5FH)
  00446 FE60              CP      60H             ;Value in range ?
  00448 3013              JR      NC,045DH        ;No: continue at 045DH

  0044A CB08              RRC     B               ;<SHIFT> pressed ?
  0044C 3031              JR      NC,047FH        ;No: key code is ok

  0044E C620              ADD     A,20H           ;Add 20H to get lower case
  00450 57                LD      D,A             ;Save value
  00451 3A40F8            LD      A,(0F840H)      ;Scan keyboard
  00454 E610              AND     10H             ;<SHIFT> + down arrow pressed ?
  00456 7A                LD      A,D             ;Restore value
  00457 2826              JR      Z,047FH         ;No: key code is ok

  00459 D660              SUB     60H             ;Convert value to control code
  0045B 1822              JR      047FH           ;Continue at 047FH

; Numbers, special characters or control characters were pressed:
; A has a value between 60H (for '0') and 77H (for space bar)

  0045D D670              SUB     70H             ;Control character pressed ?
  0045F 3010              JR      NC,0471H        ;Yes: continue at 0471H

  00461 C640              ADD     A,40H           ;Adjust code
```

```
  00463 FE3C            CP      3CH             ;',' '-' '.' or '/' pressed ?
  00465 3802            JR      C,0469H         ;No: correct ASCII code is
                                                ;in A, continue at 0469H
  00467 EE10            XOR     10H             ;Yes: With these keys, the
                                                ;symbols are swapped. This has
                                                ;to be corrected with XOR 10H
                                                ;(see ASCII table)
  00469 CB08            RRC     B               ;<SHIFT> pressed ?
  0046B 3012            JR      NC,047FH        ;No: pass on ASCII code

  0046D EE10            XOR     10H             ;Yes: exchange codes from
                                                :20H to 2FH with 30H to 3FH
  0046F 180E            JR      047FH

; Control key pressed
; Take ASCII code from table at 0050H

  00471 07              RLCA                    ;A = A * 2 as table offset
  00472 CB08            RRC     B               ;<SHIFT> pressed ?
  00474 3001            JR      NC,0477H        ;No: offset is ok

  00476 3C              INC     A               ;Adjust offset
  00477 215000          LD      HL,0050H        ;HL -> table with ASCII codes
  0047A 5F              LD      E,A             ;DE = offset
  0047B 1600            LD      D,00H
  0047D 19              ADD     HL,DE           ;Set pointer on key code
  0047E 7E              LD      A,(HL)          ;A = key code

; The correct ASCII code in now in A. Only the <MOD SEL> key has to be
; checked.

  0047F 211840          LD      HL,4018H        ;HL = flag byte
  00482 CB76            BIT     6,(HL)          ;<MOD SEL> active ?
                                                ;(graphic characters)
  00484 2824            JR      Z,04AAH         ;No: finish routine

  00486 FE2B            CP      2BH             ;ASCII code < 2BH
  00488 3820            JR      C,04AAH         ;Yes: these keys have no
                                                ;graphic character
  0048A FE30            CP      30H             ;ASCII code between 2BH, 2FH
  0048C 3004            JR      NC,0492H        ;No: check further

  0048E D62B            SUB     2BH             ;Yes: subtract 2BH
  00490 1816            JR      04A8H           ;and add C0H so that '+' gets
                                                ;code C0H and '/' gets code
                                                ;C4H

  00492 FE3B            CP      3BH             ;Code smaller then ':' ?
  00494 3814            JR      C,04AAH         ;Yes: no graphic character

  00496 FE5B            CP      5BH             ;Code bigger then 'Z' ?
  00498 3004            JR      NC,049EH        ;Yes: clear <F1> - <F4>

  0049A D636            SUB     36H             ;':' to 'Z' becomes 05H to 24H
  0049C 180A            JR      04A8H           ;Add C0H for graphic char.

  0049E FE60            CP      60H             ;<F1> - <F4> ?
  004A0 3808            JR      C,04AAH         ;Yes: finish routine

  004A2 FE7B            CP      7BH             ;<SHIFT>+<F1> - <SHIFT>+<F4> ?
  004A4 3004            JR      NC,04AAH        ;Yes: finish routine

  004A6 D63B            SUB     3BH             ;''' to 'z' becomes 25H to 3FH
  004A8 C6C0            ADD     A,0C0H          ;Add C0H for graphic char.

; Now A contains the right ASCII code from 01H to FFH

  004AA 322440          LD      (4024H),A       ;Save ASCII code for repeat
  004AD 57              LD      D,A             ;Save for delay loop
  004AE 010020          LD      BC,2000H        ;Call delay loop for
  004B1 CD6000          CALL    0060H           ;debouncing purposes
```

```
  004B4 7A              LD      A,D             ;Restore ASCII code

  004B5 FE0D            CP      0DH             ;<RETURN> pressed ?
  004B7 2807            JR      Z,04C0H         ;Yes: deactivate <MOD SEL>

  004B9 FE01            CP      01H             ;<BREAK pressed ?
  004BB 2803            JR      Z,04C0H         ;Yes: deactivate <MOD SEL>

  004BD C0              RET     NZ              ;Done when <BREAK> not pressed

  004BE EF              RST     28H             ;RST 28H when <BREAK> pressed
                                                ;(used by DOS)
  004BF C9              RET

; Deactivate <MOD SEL>

  004C0 211840          LD      HL,4018H        ;HL -> flag byte
  004C3 CBB6            RES     6,(HL)          ;Clear MOD SEL flag
  004C5 FE01            CP      01H             ;<BREAK> pressed ?
  004C7 18F4            JR      04BDH           ;Continue at 04DBH

; <MOD SEL> pressed

  004C9 3E40            LD      A,40H           ;Set bit 6 (MOD SEL flag)
  004CB AE              XOR     (HL)            ;Toggle MOD SEL flag
  004CC 77              LD      (HL),A          ;Save it in system RAM
  004CD AF              XOR     A               ;Return no value
  004CE 18DD            JR      04ADH           ;Finish routine

; <CTRL> pressed

  004D0 CBFE            SET     7,(HL)          ;Set CTRL flag
  004D2 18F9            JR      04CDH           ;Pass no value

; <RPT> pressed

  004D4 3A2440          LD      A,(4024H)       ;A = last character code
  004D7 18A6            JR      047FH           ;Use it for further processing

; Compute POS (continuation of 0350H)

  004D9 ED52            SBC     HL,DE           ;HL = screen position of cursor
  004DB 112800          LD      DE,0028H        ;DE = number of chars/line
  004DE B7              OR      A               ;Clear C-flag

  004DF ED52            SBC     HL,DE           ;Subtract 40 from HL
  004E1 30FB            JR      NC,04DEH        ;until HL < 0

  004E3 19              ADD     HL,DE           ;Adjust HL: HL = cursor
                                                ;position within line
  004E4 7D              LD      A,L             ;A = POS
  004E5 E1              POP     HL              ;Restore PTP
  004E6 C9              RET


; Printer routine (activated by DCB call)

  004E7 79              LD      A,C             ;A = ASCII code of character
                                                ;to be printed.
  004E8 B7              OR      A               ;Null character ?
  004E9 283E            JR      Z,0529H         ;Yes: Just wait for printer

  004EB FE0B            CP      0BH             ;Vertical Tab character ?
  004ED 280A            JR      Z,04F9H         ;Yes: incorporate line counter

  004EF FE0C            CP      0CH             ;Form Feed character ?
  004F1 201B            JR      NZ,050EH        ;No: Print character

  004F3 AF              XOR     A               ;A = maximum number of lines
  004F4 DDB603          OR      (IX+03H)        ;per page. A = 0 ?
  004F7 2815            JR      Z,050EH         ;Yes: send 00H to printer
```

```
  004F9 DD7E03          LD      A,(IX+03H)      ;A = max. number of lines/page
  004FC DD9604          SUB     (IX+04H)        ;minus number of already
                                                ;printed lines.
  004FF 47              LD      B,A             ;B = number of lines remaining
                                                ;until next page
  00500 CD2905          CALL    0529H           ;Printer ready ?
  00503 20FB            JR      NZ,0500H        ;No: wait for printer

  00505 0E0A            LD      C,0AH           ;C = Line Feed (0AH)
  00507 CD3C05          CALL    053CH           ;Output C to printer
  0050A 10F4            DJNZ    0500H           ;Until next line

  0050C 1816            JR      0524H           ;Line counter = 0

; Print character

  0050E CD2905          CALL    0529H           ;Printer ready ?
  00511 20FB            JR      NZ,050EH        ;No: wait for printer

  00513 CD3C05          CALL    053CH           ;Print character in C
  00516 FE0D            CP      0DH             ;was it a Carriage Return
  00518 C0              RET     NZ              ;No: done

  00519 DD3404          INC     (IX+04H)        ;Increment line counter
  0051C DD7E04          LD      A,(IX+04H)      ;A = number of printed lines
  0051F DDBE03          CP      (IX+03H)        ;Page full ?
  00522 79              LD      A,C             ;Printed character back in A
  00523 C0              RET     NZ              ;No: done

  00524 DD360400        LD      (IX+04H),00H    ;Line counter = 0
  00528 C9              RET


; Printer ready ? (AF)
;
; I: -
; O: Z-flag = 1 when printer is ready

  00529 3E07            LD      A,07H           ;Select register 7
  0052B D3F8            OUT     (0F8H),A        ;of the PSG
  0052D 3E7F            LD      A,7FH           ;Set port A to output
  0052F D3F9            OUT     (0F9H),A        ;and port B to input
  00531 3E0F            LD      A,0FH           ;Select port B
  00533 D3F8            OUT     (0F8H),A
  00535 DBF9            IN      A,(0F9H)        ;Get printer status
  00537 E6EF            AND     0EFH            ;Mask status bits
  00539 FE2F            CP      2FH             ;and set Z-flag
  0053B C9              RET

; C to printer without checking

  0053C 3E07            LD      A,07H           ;Select register 7
  0053E D3F8            OUT     (0F8H),A        ;of the PSG
  00540 3E7F            LD      A,7FH           ;I/O port A on output and
  00542 D3F9            OUT     (0F9H),A        ;I/O port B on input
  00544 3E0E            LD      A,0EH           ;Select register 14 (port A)
  00546 D3F8            OUT     (0F8H),A
  00548 79              LD      A,C             ;Output byte
  00549 D3F9            OUT     (0F9H),A
  0054B 3E07            LD      A,07H           ;Put both ports on output
  0054D D3F8            OUT     (0F8H),A
  0054F 3EFF            LD      A,0FFH
  00551 D3F9            OUT     (0F9H),A
  00553 3E0F            LD      A,0FH           ;To port B
  00555 D3F8            OUT     (0F8H),A
  00557 AF              XOR     A               ;Output 00H
  00558 D3F9            OUT     (0F9H),A
  0055A 3E0F            LD      A,0FH           ;Then to port B
  0055C D3F8            OUT     (0F8H),A
  0055E 3E01            LD      A,01H           ;Ouput 01H and indicate to
```

```
  00560 D3F9             OUT     (0F9H),A        ;printer that a byte has been
                                                 ;send
  00562 79               LD      A,C             ;Character back in A
  00563 C9               RET


; Output routine (continuation of 0335H)
                                                 ;Output to cassette ?
  00564 FA1F02           JP      M,021FH         ;Yes: continue at 021FH
                                                 ;Output to printer ?
  00567 C29C03           JP      NZ,039CH        ;Yes: continue at 039CH

  0056A C33A03           JP      033AH           ;Output to screen


; Start 2 (continuation of 06ACH)

  0056D 31F841           LD      SP,41F8H        ;Set stack pointer
  00570 AF               XOR     A               ;Clear port 255 (NBGRD, CHAR 1,
  00571 D3FF             OUT     (0FFH),A        ;and LGR)

  00573 2100F4           LD      HL,0F400H       ;Clear colour memory
  00576 1101F4           LD      DE,0F401H
  00579 01FF03           LD      BC,03FFH
  0057C 3600             LD      (HL),00H
  0057E EDB0             LDIR

  00580 0EFF             LD      C,0FFH          ;C = port address
  00582 ED78             IN      A,(C)           ;Parameters for NTSC standard
  00584 E608             AND     08H             ;(have no effect in PAL)
  00586 47               LD      B,A
  00587 D3FC             OUT     (0FCH),A
  00589 ED78             IN      A,(C)
  0058B E608             AND     08H

  0058D 211137           LD      HL,3711H        ;--
  00590 110038           LD      DE,3800H        ;DE -> CRTC table for PAL
                                                 ;standard
  00593 A8               XOR     B
  00594 00               NOP
  00595 00               NOP
  00596 D3FD             OUT     (0FDH),A
  00598 ED78             IN      A,(C)
  0059A E608             AND     08H
  0059C 216935           LD      HL,3569H        ;HL -> colour code table
  0059F A8               XOR     B
  005A0 1812             JR      05B4H           ;Continue at 05B4H


; Unused routine (only for computers with NTSC standard)

  005A2 D3FE             OUT     (0FEH),A
  005A4 ED78             IN      A,(C)
  005A6 E608             AND     08H
  005A8 210A37           LD      HL,370AH
  005AB A8               XOR     B
  005AC 2006             JR      NZ,05B4H
  005AE 213137           LD      HL,3731H
  005B1 112338           LD      DE,3823H


; Start 3 (continuation of 05A0H)

  005B4 E5               PUSH    HL              ;Save HL
  005B5 EB               EX      DE,HL           ;HL -> CRTC table
  005B6 11F042           LD      DE,42F0H        ;DE -> system RAM
  005B9 012300           LD      BC,0023H        ;BC = table size (35 bytes)
  005BC EDB0             LDIR                    ;Copy table into sytem RAM

  005BE E1               POP     HL              ;Restore HL
  005BF 119043           LD      DE,4390H        ;DE -> colour code table
```

```
  005C2 011000          LD      BC,0010H        ;BC = table size (16 colours)
  005C5 EDB0            LDIR                    ;Copy into system RAM

  005C7 C36C00          JP      006CH           ;Continue at 006CH


; Reset 2 (continuation of 0069H)

  005CA AF              XOR     A               ;A = 0;
  005CB D3ED            OUT     (0EDH),A        ;
  005CD 3A04F8          LD      A,(0F804H)      ;Read keyboard
  005D0 CB57            BIT     2,A             ;<R> pressed ?
  005D2 C20000          JP      NZ,0000H        ;Yes: cold start

  005D5 C3C006          JP      06C0H           ;No: warm start

  005D8 FF              RST     38H             ;--


; Input of one line incl. cursor control, FKEY evaluation und presentation
; of the entered character on screen (AF, BC, DE, HL).
; The input is terminated by <RETURN> or <BREAK>.
; The End Of Line is indicated in the buffer by a 0DH character
;
; I: HL -> input buffer to store the entered characters.
;    B = Maximum number of characters allowed to be entered.
;        (more characters are not accepted. The input must be terminated
;        by <RETURN> or <BREAK>)
; O: HL -> start of input buffer
;    DE = 401DH (from 0033H call)
;    B  = number of entered characters
;    C  = B from start
;    A  = last character entered (0DH or 01H)
;    C-flag = 1 when input was terminated with <BREAK>

  005D9 E5              PUSH    HL              ;Save buffer address
  005DA 3E0E            LD      A,0EH           ;A = Cursor on
  005DC CD3300          CALL    0033H           ;Output character
  005DF 48              LD      C,B             ;C = max. number of characters
  005E0 C30030          JP      3000H           ;Continue at 3000H for FKEY
                                                ;evaluation

  005E3 FE20            CP      20H             ;Control key entered ?
  005E5 3025            JR      NC,060CH        ;No: store character

  005E7 FE0D            CP      0DH             ;<RETURN> ?
  005E9 CA6206          JP      Z,0662H         ;Yes: continue at 0662H

  005EC FE1F            CP      1FH             ;<CLEAR> ?
  005EE 2829            JR      Z,0619H         ;Yes: continue at 0619H

  005F0 FE01            CP      01H             ;<BREAK> ?
  005F2 286D            JR      Z,0661H         ;Yes: continue at 0661H

  005F4 11E005          LD      DE,05E0H        ;Set new return address
  005F7 D5              PUSH    DE              ;to 05E0H

  005F8 FE08            CP      08H             ;Backspace (arrow left) ?
  005FA 2834            JR      Z,0630H         ;Yes: continue at 0630H

  005FC FE18            CP      18H             ;Delete line
                                                ;(shift-arrow left) ?
  005FE 282B            JR      Z,062BH         ;Yes: continue at 062BH

  00600 FE09            CP      09H             ;TAB (arrow right) ?
  00602 2842            JR      Z,0646H         ;Yes: continue at 0646H

  00604 FE19            CP      19H             ;Shift-arrow right ? (32
                                                ;characters/line in GENIE I)
  00606 2839            JR      Z,0641H         ;Yes: continue at 0641H
```

```
  00608 FE0A            CP      0AH             ;New Line (arrow down) ?
  0060A C0              RET     NZ              ;No: control character has no
                                                ;function, get new character

  0060B D1              POP     DE              ;Remove return address from
                                                ;stack
  0060C 77              LD      (HL),A          ;Store character in buffer
  0060D 78              LD      A,B             ;Entering of characters
  0060E B7              OR      A               ;still allowed ?
  0060F 28CF            JR      Z,05E0H         ;No: get new character

  00611 7E              LD      A,(HL)          ;Get character from buffer
  00612 23              INC     HL              ;Buffer pointer + 1
  00613 CD3300          CALL    0033H           ;Output character
  00616 05              DEC     B               ;Character counter - 1
  00617 18C7            JR      05E0H           ;Get next character

; <CLEAR> pressed

  00619 CDC901          CALL    01C9H           ;CLS
  0061C 41              LD      B,C             ;Restart character counter
  0061D E1              POP     HL              ;HL -> start of buffer
  0061E E5              PUSH    HL              ;Save pointer again
  0061F C3E005          JP      05E0H           ;Get next character

; Delete line

  00622 CD3006          CALL    0630H           ;Delete character
  00625 2B              DEC     HL              ;Buffer pointer - 1
  00626 7E              LD      A,(HL)          ;A = last character
  00627 23              INC     HL              ;Increment buffer pointer
  00628 FE0A            CP      0AH             ;Has a new line already been
                                                ;started ?
  0062A C8              RET     Z               ;Yes: then only delete until
                                                ;here
; <Shift>+<arrow left> pressed

  0062B 78              LD      A,B             ;Have characters been
  0062C B9              CP      C               ;entered already ?
  0062D 20F3            JR      NZ,0622H        ;Yes: delete line

  0062F C9              RET

; <arrow left> pressed

  00630 78              LD      A,B             ;Character counter still on
  00631 B9              CP      C               ;initial value ?
  00632 C8              RET     Z               ;Yes: done

  00633 2B              DEC     HL              ;Buffer pointer - 1
  00634 7E              LD      A,(HL)          ;Get previous character
  00635 FE0A            CP      0AH             ;New line started ?
  00637 23              INC     HL              ;Buffer pointer + 1
  00638 C8              RET     Z               ;Yes: return
  00639 2B              DEC     HL              ;No: buffer pointer + 1
  0063A 3E08            LD      A,08H           ;A = backspace character
  0063C CD3300          CALL    0033H           ;Output byte
  0063F 04              INC     B               ;Character counter + 1
  00640 C9              RET

; <SHIFT>+<arrow right> pressed

  00641 3E17            LD      A,17H           ;17H was code for switching to
                                                ;32 characters/lines
  00643 C33300          JP      0033H           ;Output byte
                                                ;(has no effect on the
                                                ;Colour Genie)
; <arrow right> pressed

  00646 CD4803          CALL    0348H           ;A = POS
  00649 E607            AND     07H             ;Calculate last TAB position
```

```
  0064B 2F              CPL                     ;Negate value
  0064C 3C              INC     A               ;adjust to 2 complement
  0064D C608            ADD     A,08H           ;+8 gives number of characters
                                                ;until next TAB position
  0064F 5F              LD      E,A             ;E = counter

  00650 78              LD      A,B             ;Is it allowed to enter more
  00651 B7              OR      A               ;characters ?
  00652 C8              RET     Z               ;No: return

  00653 3E20            LD      A,20H           ;A = space character
  00655 77              LD      (HL),A          ;Put it in buffer
  00656 23              INC     HL              ;Buffer pointer + 1
  00657 D5              PUSH    DE              ;Save DE
  00658 CD3300          CALL    0033H           ;Output byte
  0065B D1              POP     DE              ;Restore DE
  0065C 05              DEC     B               ;Character counter - 1
  0065D 1D              DEC     E               ;TAB counter - 1
  0065E C8              RET     Z               ;Return when done
  0065F 18EF            JR      0650H           ;Next character in buffer

; <BREAK> pressed

  00661 37              SCF                     ;Set C-flag = 1 and treat it
                                                ;further like <RETURN>
; <RETURN> pressed

  00662 F5              PUSH    AF              ;Save flags
  00663 3E0D            LD      A,0DH           ;A = Carriage Return
  00665 77              LD      (HL),A          ;Put it in buffer
  00666 CD3300          CALL    0033H           ;Output byte
  00669 3E0F            LD      A,0FH           ;A = cursor off
  0066B CD3300          CALL    0033H           ;Output character
  0066E 79              LD      A,C             ;Calculate the number of
  0066F 90              SUB     B               ;entered characters
  00670 47              LD      B,A             ;Result in B
  00671 F1              POP     AF              ;Restore flags
  00672 E1              POP     HL              ;HL -> start of buffer
  00673 C9              RET


; Start 1

  00674 3E08            LD      A,08H           ;CHAR 2, NBGRD and LGR
  00676 D3FF            OUT     (0FFH),A        ;Output to port
  00678 321C43          LD      (431CH),A       ;Save port status

  0067B 21D206          LD      HL,06D2H        ;HL -> start of block
  0067E 110040          LD      DE,4000H        ;DE -> destination system RAM
  00681 013600          LD      BC,0036H        ;BC = 54 bytes
  00684 EDB0            LDIR                    ;Initialize system RAM

  00686 3D              DEC     A               ;Repeat until A = 00H
  00687 3D              DEC     A
  00688 20F1            JR      NZ,067BH

  0068A 0627            LD      B,27H           ;Clear the next 39 bytes
  0068C 12              LD      (DE),A
  0068D 13              INC     DE
  0068E 10FC            DJNZ    068CH

  00690 21AB34          LD      HL,34ABH        ;HL -> function key defaults
  00693 115043          LD      DE,4350H        ;DE -> function key RAM space
  00696 013800          LD      BC,0038H        ;BC = size to copy
  00699 EDB0            LDIR                    ;Copy function key definitions

  0069B 3E01            LD      A,01H           ;Set SCALE = 1
  0069D 321443          LD      (4314H),A
  006A0 213039          LD      HL,3930H        ;Set pointer on table with
  006A3 228C43          LD      (438CH),HL      ;Colour BASIC statements in
                                                ;system RAM
```

```
  006A6 21DB39          LD      HL,39DBH          ;Set pointer on vector table
  006A9 228E43          LD      (438EH),HL        ;in system RAM
  006AC C36D05          JP      056DH             ;Continue at 05D6H

  006AF 3A00C0          LD      A,(0C000H)        ;Take first byte of ROM
                                                  ;cartridge space
  006B2 FE43            CP      'C'               ;Program there ?
  006B4 CA01C0          JP      Z,0C001H          ;Yes: execute program

  006B7 3A00C0          LD      A,(0C000H)        ;Take first byte of ROM
                                                  ;cartridge space
  006BA FE44            CP      'D'               ;Program there ?
  006BC CA01C0          JP      Z,0C001H          ;Yes: execute program

  006BF C9              RET


  006C0 3A00C0          LD      A,(0C000H)        ;Take first byte of ROM
                                                  ;cartridge space
  006C3 FE43            CP      'C'               ;Program there ?
  006C5 CA01C0          JP      Z,0C001H          ;Yes: execute program

  006C8 C3AE19          JP      19AEH


; Unused ROM space

  006CB FF              RST     38H               ;--
  006CC FF              RST     38H
  006CD FF              RST     38H
  006CE FF              RST     38H
  006CF FF              RST     38H
  006D0 FF              RST     38H
  006D1 FF              RST     38H

; The following block of code is copied into system RAM at 4000H to initialize
; the vectors and DCBs located there

  006D2 C3961C          JP      1C96H             ;RST 08H vector
  006D5 C3781D          JP      1D78H             ;RST 10H vector
  006D8 C3901C          JP      1C90H             ;RST 18H vector
  006DB C3D925          JP      25D9H             ;RST 20H vector
  006DE C9              RET                       ;RST 28H vector (BREAK vector)
  006DF 00              NOP
  006E0 00              NOP
  006E1 C9              RET                       ;RST 30H vector
  006E2 00              NOP
  006E3 00              NOP
  006E4 FB              EI                        ;RST 38H vector
  006E5 C9              RET
  006E6 00              NOP

  006E7 01              DEFB    01H               ;Initialization data for
  006E8 E303            DEFW    03E3H             ;contructing keyboard DCB
  006EA 00              DEFB    00H
  006EB 07              DEFB    07H
  006EC 40              DEFB    40H
  006ED 20              DEFB    20H
  006EE 49              DEFB    47H

  006EF 07              DEFB    07H               ;Initialization data for
  006F0 E430            DEFW    30E4H             ;contructing screen DCB
  006F2 0044            DEFW    4400H
  006F4 01              DEFB    01H
  006F4 01              DEFB    01H
  006F4 03              DEFB    01H

  006F7 06              DEFB    06H               ;Initialization data for
  006F8 E704            DEFW    04E7H             ;contructing printer DCB
  006FA 43              DEFB    43H
  006FB 00              DEFB    00H
```

```
  006FC 00              DEFB    00H
  006FD 50              DEFB    50H
  006FE 52              DEFB    52H

  006FF C30050          JP      5000H           ;
  00702 C7              RST     00H             ;
  00703 00              NOP
  00704 00              NOP

  00705 3E00            LD      A,00H           ;False DCB vector
  00707 C9              RET

; X = X + 0.5 (SNG)

  00708 218013          LD      HL,1380H        ;HL -> Constant 0.5

; X = X + (HL)

  0070B CDC209          CALL    09C2H           ;BCDE = (HL)
  0070E 1806            JR      0716H           ;X = X + BCDE

; X = (HL) - X (SNG)

  00710 CDC209          CALL    09C2H           ;BCDE = (HL)

; X = BCDE - X (SNG)

  00713 CD8209          CALL    0982H           ;X = -X

; X = BCDE + X (SNG)

  00716 78              LD      A,B             ;A = Exp (BCDE)
  00717 B7              OR      A               ;BCDE = 0 ?
  00718 C8              RET     Z               ;Yes: result already in X

  00719 3A2441          LD      A,(4124H)       ;A = Exp (X)
  0071C B7              OR      A               ;X = 0 ?
  0071D CAB409          JP      Z,09B4H         ;Yes: X = BCDE

  00720 90              SUB     B               ;A = Exp (X) - Exp (BCDE)
                                                ;Largest value in X ?
  00721 300C            JR      NC,072FH        ;Yes: continue at 072FH

  00723 2F              CPL                     ;A = -A
  00724 3C              INC     A               ;A is the difference of the
                                                ;exponents
  00725 EB              EX      DE,HL           ;Save DE
  00726 CDA409          CALL    09A4H           ;(SP) = X
                                                ;(2nd operand on stack)
  00729 EB              EX      DE,HL           ;Restore DE
  0072A CDB409          CALL    09B4H           ;X = BCDE (1st operand)
  0072D C1              POP     BC              ;BCDE = 2nd operand
  0072E D1              POP     DE
                                                ;X is now the operand with
                                                ;the biggest exponent
                                                ;A contains the differnce
                                                ;between the exponents
  0072F FE19            CP      19H             ;Difference bigger then 24 ?
                                                ;(difference between the
                                                ;operands bigger then 2^24)
  00731 D0              RET     NC              ;Yes: BCDE is too small, the
                                                ;sum would not change X

  00732 F5              PUSH    AF              ;Save exponent difference
  00733 CDDF09          CALL    09DFH           ;Adjust mantissas
  00736 67              LD      H,A             ;H, 7 = 0, if signs are not
                                                ;equal
  00737 F1              POP     AF              ;Restore exponent difference
  00738 CDD707          CALL    07D7H           ;Shifting CDE to the right by
                                                ;A bits: bring mantissa of BCDE
                                                ;on the same exponent as X
```

```
  0073B B4              OR      H               ;A,7 = 0 when unequal signs
  0073C 212141          LD      HL,4121H        ;HL -> X
                                                ;Signs equal ?
  0073F F25407          JP      P,0754H         ;No: continue at 0754H

  00742 CDB707          CALL    07B7H           ;Add mantissas
                                                ;Overflow ?
  00745 D29607          JP      NC,0796H        ;No: continue at 0796H

  00748 23              INC     HL              ;HL -> Exp (X)
  00749 34              INC     (HL)            ;Exp (X) + 1 (adjust Exp
                                                ;for overflow)
                                                ;Exponent overflow ?
  0074A CAB207          JP      Z,07B2H         ;Yes: ?OV Error

  0074D 2E01            LD      L,01H           ;Divide Mantissa by 2
  0074F CDEB07          CALL    07EBH           ;because Exponent incremented
  00752 1842            JR      0796H           ;Round CDE and store in X

; Unequal signs

  00754 AF              XOR     A               ;B = -B
  00755 90              SUB     B
  00756 47              LD      B,A
  00757 7E              LD      A,(HL)          ;Subtract Mantissa
  00758 9B              SBC     A,E
  00759 5F              LD      E,A
  0075A 23              INC     HL
  0075B 7E              LD      A,(HL)
  0075C 9A              SBC     A,D
  0075D 57              LD      D,A
  0075E 23              INC     HL
  0075F 7E              LD      A,(HL)          ;Subtract MSBs
  00760 99              SBC     A,C
  00761 4F              LD      C,A             ;Underflow ?

; SFLOAT: Convert CDEB in 2-exp format and store in X
;         The mantissa is shifted to the left by one bit until
;         the highest bit = 1 and the exponent is as small as possible

  00762 DCC307          CALL    C,07C3H         ;Invert Mantissa (CDEB) when
                                                ;over or underflow
  00765 68              LD      L,B             ;HL = LSBs
  00766 63              LD      H,E             ;Manstissa now in CDHL
  00767 AF              XOR     A               ;A = 00H
  00768 47              LD      B,A             ;B = counter
  00769 79              LD      A,C             ;A = MSB
  0076A B7              OR      A               ;MSB = 0 ?
  0076B 2018            JR      NZ,0785H        ;No: continue shifting

  0076D 4A              LD      C,D             ;C << D << H << L << 00H
  0076E 54              LD      D,H             ;(Shift Mantissa left bytewise)
  0076F 65              LD      H,L             ;until MSB <> 0
  00770 6F              LD      L,A
  00771 78              LD      A,B
  00772 D608            SUB     08H             ;Counter - 8 (8 bits shifted)
  00774 FEE0            CP      0E0H            ;Counter = -32 ?
  00776 20F0            JR      NZ,0768H        ;No: continue shifting

                                                ;Shifted 4 times:
                                                ;Mantissa is now 0, result = 0

; X = 0 (SNG)

  00778 AF              XOR     A               ;A = 00H
  00779 322441          LD      (4124H),A       ;Exp (X) = 00H,  X = 0
  0077C C9              RET

; Shift Mantissa CDHL bitwise to the left until highest bit = 1

  0077D 05              DEC     B               ;Counter - 1 (shift 1 bit)
```

```
  0077E 29              ADD     HL,HL           ;HL = HL * 2 (= shift left)
  0077F 7A              LD      A,D
  00780 17              RLA                     ;D = D * 2, C-flag = overflow
  00781 57              LD      D,A
  00782 79              LD      A,C             ;C = C + C + C-flag
  00783 8F              ADC     A,A
  00784 4F              LD      C,A
                                                ;Highest bit of C = 1 ?
  00785 F27D07          JP      P,077DH         ;No: continue shifting

  00788 78              LD      A,B             ;A = negative number of times
                                                ;that has been shifted
  00789 5C              LD      E,H             ;Mantissa = CDEB
  0078A 45              LD      B,L
  0078B B7              OR      A               ;Anything shifted ?
  0078C 2808            JR      Z,0796H         ;Yes: continue at 0796H

  0078E 212441          LD      HL,4124H        ;HL -> Exp (X)
  00791 86              ADD     A,(HL)          ;A = new Exponent
  00792 77              LD      (HL),A          ;Save Exponent
                                                ;Old Exponent to small ?
  00793 30E3            JR      NC,0778H        ;Yes: X = 0
                                                ;Exponent = 0 ?
  00795 C8              RET     Z               ;Yes: return

; Round CDEB and store in X
; Exp (X) and sign are kept

  00796 78              LD      A,B             ;A = LSB

  00797 212441          LD      HL,4124H        ;HL -> Exp (X)
  0079A B7              OR      A               ;Highest bit of LSB = 1 ?
  0079B FCA807          CALL    M,07A8H         ;Yes: round up CDE

  0079E 46              LD      B,(HL)          ;B = Exp (X)
  0079F 23              INC     HL
  007A0 7E              LD      A,(HL)          ;A = Sign flag
  007A1 E680            AND     80H             ;Clear sign bit
  007A3 A9              XOR     C
  007A4 4F              LD      C,A             ;BCDE is negative in case
                                                ;sign flag was 00H
  007A5 C3B409          JP      09B4H           ;X = BCDE

; Round CDE and write overflow to Exp (X)

  007A8 1C              INC     E               ;LSB + 1
                                                ;Overflow ?
  007A9 C0              RET     NZ              ;No: return

  007AA 14              INC     D               ;Next byte + 1
                                                ;Overflow ?
  007AB C0              RET     NZ              ;No: return

  007AC 0C              INC     C               ;MSB + 1
                                                ;Overflow ?
  007AD C0              RET     NZ              ;No: return

  007AE 0E80            LD      C,80H           ;Set highest bit of Mantissa
  007B0 34              INC     (HL)            ;and increment Exp (X)
                                                ;Overflow ?
  007B1 C0              RET     NZ              ;No: return

; ?OV Error

  007B2 1E0A            LD      E,0AH           ;E = error code for ?OV error
  007B4 C3A219          JP      19A2H           ;Continue at error routine

; CDE = CDE + (HL) (SNG)
; Fixed point (Mantissa) addition

  007B7 7E              LD      A,(HL)          ;A = (HL)
```

```
  007B8 83              ADD     A,E             ;A = E + (HL)
  007B9 5F              LD      E,A             ;E = E + (HL)
  007BA 23              INC     HL              ;Next byte
  007BB 7E              LD      A,(HL)
  007BC 8A              ADC     A,D             ;Now add with carry
  007BD 57              LD      D,A
  007BE 23              INC     HL              ;Last byte
  007BF 7E              LD      A,(HL)
  007C0 89              ADC     A,C
  007C1 4F              LD      C,A
  007C2 C9              RET

; Invert Mantissa of CDEB and Sign flag (4125H)

  007C3 212541          LD      HL,4125H        ;HL -> Sign flag
  007C6 7E              LD      A,(HL)          ;A = Sign Flag
  007C7 2F              CPL                     ;Invert
  007C8 77              LD      (HL),A          ;and store

  007C9 AF              XOR     A               ;A = 00H
  007CA 6F              LD      L,A             ;L = 00H
  007CB 90              SUB     B
  007CC 47              LD      B,A             ;B = 00H - B
  007CD 7D              LD      A,L             ;A = 00H
  007CE 9B              SBC     A,E             ;E = 00H - E - C-flag
  007CF 5F              LD      E,A
  007D0 7D              LD      A,L             ;Same with D
  007D1 9A              SBC     A,D
  007D2 57              LD      D,A
  007D3 7D              LD      A,L             ;and with C
  007D4 99              SBC     A,C
  007D5 4F              LD      C,A
  007D6 C9              RET

; Shift CDE by A bits to the right. B becomes LSB

  007D7 0600            LD      B,00H           ;Overflow on 00H

  007D9 D608            SUB     08H             ;More as 8 shifts ?
  007DB 3807            JR      C,07E4H         ;No: shift bitwise
                                                ;Yes: shift bytewise
  007DD 43              LD      B,E
  007DE 5A              LD      E,D
  007DF 51              LD      D,C
  007E0 0E00            LD      C,00H           ;00H >> C >> D >> E >> B
  007E2 18F5            JR      07D9H           ;Loop

; Shift bitwise

  007E4 C609            ADD     A,09H           ;Reverse SUB and adjust by 1
  007E6 6F              LD      L,A             ;L = counter

  007E7 AF              XOR     A               ;A = 00H
  007E8 2D              DEC     L               ;Counter - 1
                                                ;Counter = 0 ?
  007E9 C8              RET     Z               ;Yes: return

  007EA 79              LD      A,C             ;Shift right CDE by one bit
  007EB 1F              RRA
  007EC 4F              LD      C,A
  007ED 7A              LD      A,D             ;Shift D
  007EE 1F              RRA
  007EF 57              LD      D,A
  007F0 7B              LD      A,E             ;Shift E
  007F1 1F              RRA
  007F2 5F              LD      E,A
  007F3 78              LD      A,B             ;Shift B
  007F4 1F              RRA
  007F5 47              LD      B,A             ;Overflow in B
  007F6 18EF            JR      07E7H           ;Loop
```

```
; Constant 1 (SNG)

  007F8 00              DEFB    00H
  007F9 00              DEFB    00H
  007FA 00              DEFB    00H
  007FB 81              DEFB    81H


; Table of SNG coefficients for LOG function

  007FC 03              DEFB    03H             ;3 Coefficients

  007FD AA              DEFB    AAH             ;0.598979 approx.
  007FE 56              DEFB    56H             ;2 / ( 5*LOG(2) )
  007FF 19              DEFB    19H
  00800 80              DEFB    80H

  00801 F1              DEFB    F1H             ;0.961471 approx.
  00802 22              DEFB    22H             ;2 / ( 3*LOG(2) )
  00803 76              DEFB    76H
  00804 80              DEFB    80H

  00805 45              DEFB    45H             ;2.88539  approx.
  00806 AA              DEFB    AAH             ;2 / ( 1*LOG(2) )
  00807 38              DEFB    38H
  00808 82              DEFB    82H


; X = LOG ( X )
; -------------
; Calculates the natural logarithm of X
;
; I: X = numerical value (<> 0)
; O: X = LOG (numerical value)

  00809 CD5509          CALL    0955H           ;TEST2
  0080C B7              OR      A               ;Argument = 0 ?
  0080D EA4A1E          JP      PE,1E4AH        ;Yes: ?FC Error

  00810 212441          LD      HL,4124H        ;HL -> Exp (argument)
  00813 7E              LD      A,(HL)          ;A = Exp (argument)
  00814 013580          LD      BC,8035H        ;BCDE = 0.707107 = SQR(2)/2
  00817 11F304          LD      DE,04F3H
                                                ;Conversion Arg = x * 2 ^ n
  0081A 90              SUB     B               ;Exp (Arg) - 128 = n (B is 128)
  0081B F5              PUSH    AF              ;save n
  0081C 70              LD      (HL),B          ;X = x (set Exp to 80H)
  0081D D5              PUSH    DE              ;Save BCDE
  0081E C5              PUSH    BC
  0081F CD1607          CALL    0716H           ;X = X + BCDE = X+1/2*SQR(2)
  00822 C1              POP     BC              ;Restore BCDE
  00823 D1              POP     DE
  00824 04              INC     B               ;Exp(BCDE) + 1
                                                ;BCDE = BCDE * 2 + SQR(2)
  00825 CDA208          CALL    08A2H           ;X = BCDE / X = SQR(2) / X
  00828 21F807          LD      HL,07F8H        ;HL -> 1.0
  0082B CD1007          CALL    0710H           ;X = (HL) - X = 1.0 - X
  0082E 21FC07          LD      HL,07FCH        ;HL -> numeric table
  00831 CD9A14          CALL    149AH           ;Compute row1
  00834 018080          LD      BC,8080H        ;BCDE = -0.5
  00837 110000          LD      DE,0000H
  0083A CD1607          CALL    0716H           ;X = BCDE + X = X - 0.5
  0083D F1              POP     AF              ;Restore n
  0083E CD890F          CALL    0F89H           ;X = X + n
                                                ;and multiply with LOG(2)

; X = X * LOG(2)

  00841 013180          LD      BC,8031H        ;BCDE = 0.693147 = LOG(2)
  00844 111872          LD      DE,7218H
```

```
; SMUL: X = BCDE * X
; Multiply two single precision numbers
;
; I: BCDE = 1st factor
;    X    = 2nd factor
; O: X    = product

  00847 CD5509          CALL    0955H           ;TEST2
  0084A C8              RET     Z               ;X = 0: result = 0

  0084B 2E00            LD      L,00H           ;Flag = 0 (MUL indication)
  0084D CD1409          CALL    0914H           ;Process exponent
  00850 79              LD      A,C             ;Save CDE (1st factor) in
  00851 324F41          LD      (414FH),A       ;system RAM from 414FH onwards
  00854 EB              EX      DE,HL
  00855 225041          LD      (4150H),HL
  00858 010000          LD      BC,0000H        ;BCDE = 00000000H
  0085B 50              LD      D,B
  0085C 58              LD      E,B
  0085D 216507          LD      HL,0765H        ;Put new return address
  00860 E5              PUSH    HL              ;to 0765H
  00861 216908          LD      HL,0869H        ;Put new return address twice
  00864 E5              PUSH    HL              ;to 0869H
  00865 E5              PUSH    HL
  00866 212141          LD      HL,4121H        ;HL -> 2nd factor
  00869 7E              LD      A,(HL)          ;A = next byte of mantissa of
                                                ;the 2nd factor
  0086A 23              INC     HL              ;Pointer + 1
  0086B B7              OR      A               ;Byte = 00H ?
  0086C 2824            JR      Z,0892H         ;Yes: continue at 0892H

  0086E E5              PUSH    HL              ;Save pointer
  0086F 2E08            LD      L,08H           ;L = counter for 8 bits
  00871 1F              RRA                     ;Shift next bit into C-flag
  00872 67              LD      H,A             ;Save byte in H
  00873 79              LD      A,C             ;A = MSB
                                                ;Bit set by last shift ?
  00874 300B            JR      NC,0881H        ;No: continue at 0881H

  00876 E5              PUSH    HL              ;Save HL
  00877 2A5041          LD      HL,(4150H)      ;CDE = CDE + 1st factor
  0087A 19              ADD     HL,DE
  0087B EB              EX      DE,HL
  0087C E1              POP     HL              ;Restore HL
  0087D 3A4F41          LD      A,(414FH)
  00880 89              ADC     A,C             ;A = MSB
  00881 1F              RRA                     ;CDEB one bit to the right
  00882 4F              LD      C,A
  00883 7A              LD      A,D             ;Shift D
  00884 1F              RRA
  00885 57              LD      D,A
  00886 7B              LD      A,E             ;Shift E
  00887 1F              RRA
  00888 5F              LD      E,A
  00889 78              LD      A,B             ;Shift B
  0088A 1F              RRA
  0088B 47              LD      B,A
  0088C 2D              DEC     L               ;Counter - 1
  0088D 7C              LD      A,H             ;Byte back into A
  0088E 20E1            JR      NZ,0871H        ;Check next bit

  00890 E1              POP     HL              ;HL -> X
  00891 C9              RET                     ;RET twice to 0896H and
                                                ;once to 0765H

; SUB for SMUL
; Shift CDEB 1 byte to the right and fill with 00H

  00892 43              LD      B,E             ;00H >> C >> D >> E >> B
  00893 5A              LD      E,D
```

```
  00894 51               LD      D,C
  00895 4F               LD      C,A
  00896 C9               RET


; SDIV10: X = X / 10
; Divides number in X BY 10
;
; I: X = single precision number
; O: X = number / 10

  00897 CDA409           CALL    09A4H           ;(SP) = X
  0089A 21D80D           LD      HL,0DD8H        ;HL -> 10

; X = (SP) / (HL)

  0089D CDB109           CALL    09B1H           ;X = BCDE = (HL)

; X = (SP) / X

  008A0 C1               POP     BC              ;BCDE = (SP)
  008A1 D1               POP     DE


; SDIV: X = BCDE / X
; Divides 2 single precision numbers
;
; I: BCDE = dividend
;    X = divisor
; O: X = quotient

  008A2 CD5509           CALL    0955H           ;TEST2
                                                 ;X = 0 ?
  008A5 CA9A19           JP      Z,199AH         ;Yes: ?/0 Error

  008A8 2EFF             LD      L,0FFH          ;Flag = FFH (DIV marker)
  008AA CD1409           CALL    0914H           ;Process exponent and sign
  008AD 34               INC     (HL)            ;Adjust exponents
  008AE 34               INC     (HL)
  008AF 2B               DEC     HL              ;HL -> MSB (X)
  008B0 7E               LD      A,(HL)          ;Store divisor in RAM
  008B1 328940           LD      (4089H),A       ;MSB at 4089H
  008B4 2B               DEC     HL
  008B5 7E               LD      A,(HL)
  008B6 328540           LD      (4085H),A       ;1st LSB at 4085H
  008B9 2B               DEC     HL
  008BA 7E               LD      A,(HL)
  008BB 328140           LD      (4081H),A       ;2nd LSB at 4081H
  008BE 41               LD      B,C             ;BHL = CDE
  008BF EB               EX      DE,HL           ;BHL = mantissa of dividend
  008C0 AF               XOR     A               ;A = 00H
  008C1 4F               LD      C,A             ;CDE = 000000H
  008C2 57               LD      D,A             ;(Result is computed in CDE)
  008C3 5F               LD      E,A
  008C4 328C40           LD      (408CH),A       ;(408C) = 00H
  008C7 E5               PUSH    HL              ;Save dividend
  008C8 C5               PUSH    BC
  008C9 7D               LD      A,L             ;A = 2nd LSB of dividend
  008CA CD8040           CALL    4080H           ;BHL = BHL - X (subtract
                                                 ;mantissa of dividend and
                                                 ;divisor)
                                                 ;A = (408CH)
  008CD DE00             SBC     A,00H           ;A = A - C-flag (subtract
                                                 ;borrow from last subtract
                                                 ;from MSB
  008CF 3F               CCF                     ;Invert C-flag
  008D0 3007             JR      NC,08D9H        ;Yes: continue at 08D9H

  008D2 328C40           LD      (408CH),A       ;No: write MSB back
  008D5 F1               POP     AF              ;Remove dividend from stack
  008D6 F1               POP     AF
```

```
  008D7 37                SCF                      ;C-flag = 1: jump is not
  008D8 D2C1E1            JP      NC,0E1C1H        ;executed
* 008D9   C1              POP     BC               :Dividend back to BHL
* 008DA     E1            POP     HL               ;(C-flag = 0 because of 08D0H)
  008DB 79                LD      A,C              ;A = MSB of result
  008DC 3C                INC     A                ;Bit 7 of A = 1 ?
  008DD 3D                DEC     A                ;(S-flag is influenced)
  008DE 1F                RRA                      ;Shift C-flag to bit 7 of A
                                                   ;for round routine
  008DF FA9707            JP      M,0797H          ;Yes: done, round CDE upwards
                                                   ;(if C-flag was 1) and write
                                                   ;result in X
  008E2 17                RLA                      ;Shift back and shift C-flag
                                                   ;into result
  008E3 7B                LD      A,E              ;(C << C << E << C-flag)
  008E4 17                RLA
  008E5 5F                LD      E,A
  008E6 7A                LD      A,D
  008E7 17                RLA
  008E8 57                LD      D,A
  008E9 79                LD      A,C
  008EA 17                RLA
  008EB 4F                LD      C,A
  008EC 29                ADD     HL,HL            ;Shift dividend 1 bit left
  008ED 78                LD      A,B
  008EE 17                RLA                      ;Overflow from HL to B
  008EF 47                LD      B,A
  008F0 3A8C40            LD      A,(408CH)        ;Overflow of dividend-shift
  008F3 17                RLA                      ;to 408CH (MSB of dividend)
  008F4 328C40            LD      (408CH),A
  008F7 79                LD      A,C              ;CDE = 000000H ?
  008F8 B2                OR      D
  008F9 B3                OR      E
  008FA 20CB              JR      NZ,08C7H         ;No: process next bit
                                                   ;Yes:
  008FC E5                PUSH    HL               ;Save HL
  008FD 212441            LD      HL,4124H         ;HL -> Exp (X)
  00900 35                DEC     (HL)             ;X = X / 2
                                                   ;Underflow ?
  00901 E1                POP     HL               ;Restore HL first
  00902 20C3              JR      NZ,08C7H         ;No: process next bit

  00904 C3B207            JP      07B2H            ;Yes: ?OV Error


; SUB for SMUL, SDIV, DMUL and DDIV
; Process exponents and signs

; Entry for DDIV

  00907 3EFF              LD      A,0FFH           ;Flag = FFH
* 00909 2EAF              LD      L,AFH            ;--

; Entry for DMUL

  0090A   AF              XOR     A                ;Flag = 00H
  0090B 212D41            LD      HL,412DH         ;HL -> MSB (Y)
  0090E 4E                LD      C,(HL)           ;C = MSB (Y)
  0090F 23                INC     HL               ;HL -> Exp (Y)
  00910 AE                XOR     (HL)             ;DMUL: A = Exp (Y)
                                                   ;DDIV: A = -Exp (Y) - 1
  00911 47                LD      B,A              ;B = Exp
  00912 2E00              LD      L,00H            ;Continue as with SMUL

; Entry for SMUL (L=00H) and SDIV (L=FFH)

  00914 78                LD      A,B              ;A = Exp (BCDE)
  00915 B7                OR      A                ;BCDE = 0 ?
  00916 281F              JR      Z,0937H          ;Yes: result = 0

  00918 7D                LD      A,L              ;A = flag
```

```
  00919 212441          LD      HL,4124H        ;HL -> Exp (X)
  0091C AE              XOR     (HL)            ;SMUL: A = EXP (X)
                                                ;SDIV: A = -Exp (X) - 1
  0091D 80              ADD     A,B             ;SMUL: A = Exp (Y) + Exp (X)
                                                ;SDIV: A = Exp (Y) - Exp (X) -1
  0091E 47              LD      B,A             ;B = new Exp
  0091F 1F              RRA                     ;Shift C-flag into A, bit 7
  00920 A8              XOR     B               ;XOR with B
                                                ;A, bit 7 = 0 when an overflow
                                                ;or underflow occurs to C-flag
                                                ;and A, bit 7 at the addition
                                                ;of the exponents. (if both
                                                ;exponents were positive (i.e.
                                                ;> 80H), only an oveflow to
                                                ;C-flag occurs; A, bit 7
                                                ;becomes 0! )
  00921 78              LD      A,B             ;A = new exponent
  00922 F23609          JP      P,0936H         ;Under/overflow: continue at
                                                ;0936H
  00925 C680            ADD     A,80H           ;Add offset to Exp
  00927 77              LD      (HL),A          ;Store new exponent in Exp (X)
                                                ;Exponent = 0 ?
  00928 CA9008          JP      Z,0890H         ;Yes: done, continue ar 0890H

  0092B CDDF09          CALL    09DFH           ;Adjust mantissas
  0092E 77              LD      (HL),A          ;Save signflag
  0092F 2B              DEC     HL              ;HL = 4124H
  00930 C9              RET

; Entry from Exp (X) when X > 127 or LSB (X) > 7DH
; If X < 0, then set X = 0 else ?OV Error

  00931 CD5509          CALL    0955H           ;TEST2
  00934 2F              CPL                     ;If X < 0 then A positive
                                                ;         else A negative
  00935 E1              POP     HL              ;Remove RET address
  00936 B7              OR      A               ;Set flags
  00937 E1              POP     HL              ;Remove RET address
  00938 F27807          JP      P,0778H         ;A positive then X = 0

  0093B C3B207          JP      07B2H           ;Else ?OV Error


; SMUL10: X = X * 10
; Multiplies number in X with 10
;
; I: X = single precision number
; O: X = number * 10

  0093E CDBF09          CALL    09BFH           ;BCDE = X
  00941 78              LD      A,B             ;BCDE = 0 ?
  00942 B7              OR      A
  00943 C8              RET     Z               ;Yes: result = 0

  00944 C602            ADD     A,02H           ;Exp + 2: BCDE = BCDE * 4
                                                ;Exp overflow ?
  00946 DAB207          JP      C,07B2H         ;Yes: ?OV Error

  00949 47              LD      B,A             ;Set Exp in BCDE
  0094A CD1607          CALL    0716H           ;X = X + BCDE = X + 4*X = 5 * X
  0094D 212441          LD      HL,4124H        ;HL -> Exp (X)
  00950 34              INC     (HL)            ;Exp (X) + 1: X = X * 2
                                                ;Overflow ?
  00951 C0              RET     NZ              ;No: done, return

  00952 C3B207          JP      07B2H           ;?OV Error


; TEST2: SNG function for X
; Test single or double precision number in X if smaller,
; equal or greater then zero
```

```
; ?TM Error when X contains a STR
;
; I: X = single or double precision number to be tested
; O: if X < 0: A = FFH, C-flag = 1, Z-flag = 0, S-flag = 1
; O: if X = 0: A = 00H, C-flag = 0, Z-flag = 1, S-flag = 0, P/V-flag = 1
; O: if X > 0: A = 01H, C-flag = 0, Z-flag = 0, S-flag = 0


  00955 3A2441          LD      A,(4124H)       ;A = Exp (X)
  00958 B7              OR      A               ;X = 0 ?
  00959 C8              RET     Z               :Yes: return

  0095A 3A2341          LD      A,(4123H)       ;A,7 = sign
  0095D FE2F            CP      2FH             ;--

; Entry from compare routines

* 0095D   2F            CPL
  0095F 17              RLA                     ;C-flag = sign
  00960 9F              SBC     A,A             ;A = FFH if X<0 else A = 00H
                                                ;X < 0 ?
  00961 C0              RET     NZ              ;Yes: return

  00962 3C              INC     A               ;X > 0: A = 01H
  00963 C9              RET


; FLOATA: convert A single precision number
;
; I: A = number
; O: X = numerical value of A in single precision

  00964 0688            LD      B,88H           ;B = Exp for 2 ^ 8
  00966 110000          LD      DE,0000H        ;LSBs = 0


; FLOAT: convert a binary number into single precision
; 1. Binary number of 8 bits:
;    I: B = 88H (Exp for 2 ^ 8)
;       A = binary value
;       DE = 0000H
; 2. Binary number of 16 bits: (see also CINT)
;    I: B = 90H (Exp for 2 ^ 16)
;       A = MSB of 16 bit value
;       D = LSB of 16 bit value
;       E = 0000H
; 3. Binary number of 24 bits:
;    I: B = 98H (Exp for 2 ^ 24)
;       A = MSB of 24 bit value
;       DE = LSBs of 24 bit value
; O: X = single precision number

  00969 212441          LD      HL,4124H        ;HL -> Exp (X)
  0096C 4F              LD      C,A             ;C = 8 bit value
  0096D 70              LD      (HL),B          ;Set exponent
  0096E 0600            LD      B,00H           ;B = 00H (becomes LSB at 0762H)
  00970 23              INC     HL              ;HL -> sign-flag
  00971 3680            LD      (HL),80H        ;Set sign-flag to 80H
  00973 17              RLA                     ;Sign-bit to C-flag (for 0762H)
  00974 C36207          JP      0762H           ;SFLOAT conversion CDEB to X


; X = ABS (X)
; -----------

  00977 CD9409          CALL    0994H           ;TEST1
                                                :X is positive ?
  0097A F0              RET     P               ;Yes: return

  0097B E7              RST     20H             ;TSTTYP
                                                ;INT type ?
```

```
  0097C FA5B0C           JP      M,0C5BH          ;Yes: continue at 0C5BH
                                                  ;STR type ?
  0097F CAF60A           JP      Z,0AF6H          ;Yes: ?TM error


; SDNEG: X = -X
; Negate number in X
;
; I: X = single of double precision number
; O: X = negated number

  00982 212341           LD      HL,4123H         ;HL -> MSB of X
  00985 7E               LD      A,(HL)           ;A = MSB of X
  00986 EE80             XOR     80H              ;Invert sign bit
  00988 77               LD      (HL),A           ;Store MSB of X
  00989 C9               RET


; X = SNG (X)
; -----------

  0098A CD9409           CALL    0994H            ;TEST1
  0098D 6F               LD      L,A              ;L = SNG(value) (FFH,00H,01H)
  0098E 17               RLA                      ;Highest bit to C-flag
  0098F 9F               SBC     A,A              ;A = 00 when positive result
                                                  ;else A = FFH
  00990 67               LD      H,A              ;Now HL = A
  00991 C39A0A           JP      0A9AH            ;Write HL to X as INT


; TEST1: As SNG function, but result in A
; Test number in X if smaller, equal or greater then zero
; ?TM Error when X contains a STR
;
; I: X = number to be tested
; O: if X < 0: A = FFH, C-flag = 1, Z-flag = 0, S-flag = 1
; O: if X = 0: A = 00H, C-flag = 0, Z-flag = 1, S-flag = 0, P/V-flag = 1
; O: if X > 0: A = 01H, C-flag = 0, Z-flag = 0, S-flag = 0

  00994 E7               RST     20H              ;TSTTYP
                                                  ;STR type ?
  00995 CAF60A           JP      Z,0AF6H          ;Yes: ?TM Error
                                                  ;INT type ?
  00998 F25509           JP      P,0955H          ;No: SNG or DBL so TEST2

  0099B 2A2141           LD      HL,(4121H)       ;HL = X (INT)
  0099E 7C               LD      A,H              ;HL = 0000 ?
  0099F B5               OR      L
  009A0 C8               RET     Z                ;Yes: return

  009A1 7C               LD      A,H              ;A = MSB of X
                                                  ;if H < 0 then A, bit 7 = 1
                                                  ;         else A, bit 7 = 0
  009A2 18BB             JR      095FH            :Continue inside TEST2

; (SP) = X (SNG)
; (Save value in X)

  009A4 EB               EX      DE,HL            ;Save HL
  009A5 2A2141           LD      HL,(4121H)       ;HL = LSB (X)
  009A8 E3               EX      (SP),HL          ;Put LSBs on stack
  009A9 E5               PUSH    HL               ;Save RET address
  009AA 2A2341           LD      HL,(4123H)       ;Same with MSB and Exp
  009AD E3               EX      (SP),HL
  009AE E5               PUSH    HL
  009AF EB               EX      DE,HL            ;Restore HL
  009B0 C9               RET


; X = BCDE = (HL) (SNG)
```

```
  009B1 CDC209           CALL    09C2H           ;BCDE = (HL)

; X = BCDE (SNG)

  009B4 EB               EX      DE,HL           ;HL = LSBs
  009B5 222141           LD      (4121H),HL      ;Store in X
  009B8 60               LD      H,B             ;HL = MSB and Exp
  009B9 69               LD      L,C
  009BA 222341           LD      (4123H),HL      ;Store in X
  009BD EB               EX      DE,HL           ;Restore HL
  009BE C9               RET


; BCDE = X (SNG)

  009BF 212141           LD      HL,4121H        ;HL -> X

; BCDE = (HL) (SNG)

  009C2 5E               LD      E,(HL)          ;E = LSB
  009C3 23               INC     HL
  009C4 56               LD      D,(HL)          ;D = LSB + 1
  009C5 23               INC     HL
  009C6 4E               LD      C,(HL)          ;C = MSB
  009C7 23               INC     HL
  009C8 46               LD      B,(HL)          ;B = Exp
  009C9 23               INC     HL
  009CA C9               RET

; (HL) = X (SNG)

  009CB 112141           LD      DE,4121H        ;DE -> X
  009CE 0604             LD      B,04H           ;B = size of SNG variable
  009D0 1805             JR      09D7H           ;Continue at 09D7H

; (DE) = (HL)
; Copy memory from (HL) to (DE), counter = (40AFH)

  009D2 EB               EX      DE,HL

; As above but from (DE) to (HL)

  009D3 3AAF40           LD      A,(40AFH)       ;A = variable type in X

; Copy routine: copies A bytes from (DE) to (HL)

  009D6 47               LD      B,A             ;Save in B as counter

; Copy B bytes from (DE) to (HL)

  009D7 1A               LD      A,(DE)          ;Copy byte from (DE)
  009D8 77               LD      (HL),A          ;to (HL)
  009D9 13               INC     DE              ;Origin pointer + 1
  009DA 23               INC     HL              ;Destiniation pointer + 1
  009DB 05               DEC     B               ;All bytes done ?
  009DC 20F9             JR      NZ,09D7H        ;No: next byte

  009DE C9               RET


; Sign processing for basic arithmatic operations
; Set proper mantissas of X and BCDE (highest bit to 1)
; and adjust sign
;
; I: X = single or double precision number
;    BCDE = single or double precision number
; 0: Correct mantissa of X and BCDE (highest bit = 1)
;    If both signs are the same then A, bit 7 = 1 else A, bit 7 = 0


  009DF 212341           LD      HL,4123H        ;HL -> MSB of X
```

```
  009E2 7E               LD      A,(HL)          ;A = MSB of X
  009E3 07               RLCA                    ;C-flag and A, bit 0 = Sign X
  009E4 37               SCF                     ;C-flag = 1
  009E5 1F               RRA                     ;A, bit 7 = C-flag = 1,
                                                 ;C-flag = A, bit 0 = sign
  009E6 77               LD      (HL),A          ;Correct Mantissa
  009E7 3F               CCF                     ;Invert C-flag
  009E8 1F               RRA                     ;A, bit 0 = inverted sign
  009E9 23               INC     HL
  009EA 23               INC     HL              ;HL = 4125H (sign-flag)
  009EB 77               LD      (HL),A          ;Save sign flag
  009EC 79               LD      A,C             ;Same with BCDE but leave sign
  009ED 07               RLCA
  009EE 37               SCF
  009EF 1F               RRA
  009F0 4F               LD      C,A
  009F1 1F               RRA
  009F2 AE               XOR     (HL)            ;XOR both signs
  009F3 C9               RET

; X = Y (SNG,DBL)

  009F4 212741           LD      HL,4127H        ;HL -> LSB of Y

; X = (HL) (SNG,DBL)

  009F7 11D209           LD      DE,09D2H        ;DE = address of copy routine
                                                 ;(HL) to (DE)
  009FA 1806             JR      0A02H           ;Continue at 0A02H

; Y = X (SNG,DBL)

  009FC 212741           LD      HL,4127H        ;HL -> LSB of Y

; (HL) = X (SNG,DBL)

  009FF 11D309           LD      DE,09D3H        ;DE = address of copy routine
                                                 ;(DE) to (HL)
  00A02 D5               PUSH    DE              ;Put address on stack as
                                                 ;new RET address
  00A03 112141           LD      DE,4121H        ;DE -> X (SNG)
  00A06 E7               RST     20H             ;TSTTYP
                                                 ;SNG ?
  00A07 D8               RET     C               ;Yes: ok

  00A08 111D41           LD      DE,411DH        ;DE -> X (DBL)

  00A0B C9               RET


; CP X , BCDE (SNG)
;
; I: -
; O: if X < BCDE then A = FFH, C-flag = 1, Z-flag = 0, S-flag = 1
;    if X = BCDE then A = 00H, C-flag = 0, Z-flag = 1, S-flag = 0
;    if X > BCDE then A = 01H, C-flag = 0, Z-flag = 0, S-flag = 0

  00A0C 78               LD      A,B             ;BCDE = 0 ?
  00A0D B7               OR      A
  00A0E CA5509           JP      Z,0955H         ;Yes: TEST2

  00A11 215E09           LD      HL,095EH        ;Set new RET address to TEST2
  00A14 E5               PUSH    HL
  00A15 CD5509           CALL    0955H           ;TEST2
  00A18 79               LD      A,C             ;A = MSB (BCDE)
                                                 ;X = 0 ?
  00A19 C8               RET     Z               ;Yes: goto TEST2 (only test
                                                 ;BCDE)

  00A1A 212341           LD      HL,4123H        ;HL -> MSB (X)
  00A1D AE               XOR     (HL)            ;Compare signs
```

```
  00A1E 79              LD      A,C             ;A = MSB (BCDE)
  00A1F F8              RET     M               ;Goto TEST2, when both signs
                                                ;are not equal (only test BCDE)
  00A20 CD260A          CALL    0A26H           ;Compare X and BCDE bytewise
  00A23 1F              RRA                     ;A, bit 7 = C-flag
  00A24 A9              XOR     C               ;XOR with sign of BCDE
  00A25 C9              RET                     ;Continue at TEST2


; SUB for CP
; Compare X and BCDE bytewise

  00A26 23              INC     HL              ;Compare Exp
  00A27 78              LD      A,B
  00A28 BE              CP      (HL)
  00A29 C0              RET     NZ              ;If not equal: return

  00A2A 2B              DEC     HL              :Same with MSBs
  00A2B 79              LD      A,C
  00A2C BE              CP      (HL)
  00A2D C0              RET     NZ

  00A2E 2B              DEC     HL              ;1st LSB
  00A2F 7A              LD      A,D
  00A30 BE              CP      (HL)
  00A31 C0              RET     NZ

  00A32 2B              DEC     HL              ;2nd LSB
  00A33 7B              LD      A,E
  00A34 96              SUB     (HL)            ;Subtract, so that A = 00H
                                                ;in case E = (HL)
  00A35 C0              RET     NZ

  00A36 E1              POP     HL              ;Remove RET address (0A23H)
  00A37 E1              POP     HL              ;Remove RET address (095E)
  00A38 C9              RET                     ;Return with A = 00H, Z-flag = 1


; CP HL , DE (INT)
;
; I: -
; O: if HL < DE then A = FFH, C-flag = 1, Z-flag = 0, S-flag = 1
;    if HL = DE then A = 00H, C-flag = 0, Z-flag = 1, S-flag = 0
;    if HL > DE then A = 01H, C-flag = 0, Z-flag = 0, S-flag = 0

  00A39 7A              LD      A,D             ;Compare signs
  00A3A AC              XOR     H
  00A3B 7C              LD      A,H             ;A = MSB (HL)
  00A3C FA5F09          JP      M,095FH         ;If signs are not equal:
                                                ;continue at TEST2
  00A3F BA              CP      D               ;Compare MSBs
  00A40 C26009          JP      NZ,0960H        ;If not equal: goto TEST2

  00A43 7D              LD      A,L             ;Compare LSBs
  00A44 93              SUB     E               ;Subtract, so that A = 00H
                                                ;in case E = L
  00A45 C26009          JP      NZ,0960H        ;If not equal: goto TEST2

  00A48 C9              RET


; CP X , (DE) (DBL)
;
; I: -
; O: if X < (DE) then A = FFH, C-flag = 1, Z-flag = 0, S-flag = 1
;    if X = (DE) then A = 00H, C-flag = 0, Z-flag = 1, S-flag = 0
;    if X > (DE) then A = 01H, C-flag = 0, Z-flag = 0, S-flag = 0

  00A49 212741          LD      HL,4127H        ;HL -> Y
  00A4C CDD309          CALL    09D3H           ;Copy (DE) to (HL)
  00A4F 112E41          LD      DE,412EH        ;DE -> Exp (Y)
```

```
  00A52 1A              LD      A,(DE)          ;A = Exp (Y)
  00A53 B7              OR      A               ;Y = 0 ?
  00A54 CA5509          JP      Z,0955H         ;Yes: execute TEST2

  00A57 215E09          LD      HL,095EH        ;Set new RET address
  00A5A E5              PUSH    HL              ;to TEST2
  00A5B CD5509          CALL    0955H           ;TEST2
  00A5E 1B              DEC     DE              ;DE = MSB (Y)
  00A5F 1A              LD      A,(DE)          ;A = MSB (Y)
  00A60 4F              LD      C,A             ;C = MSB (Y) (for TEST2)
                                                ;X = 0 ?
  00A61 C8              RET     Z               ;Yes: TEST2 using Y

  00A62 212341          LD      HL,4123H        ;HL -> MSB (X)
  00A65 AE              XOR     (HL)            ;Compare signs
  00A66 79              LD      A,C             ;A = MSB (Y)
                                                ;Signs not equal ?
  00A67 F8              RET     M               ;Yes: TEST2

  00A68 13              INC     DE              ;DE = Exp (Y)
  00A69 23              INC     HL              ;HL = Exp (X)
  00A6A 0608            LD      B,08H           ;B = counter for 8 bytes
  00A6C 1A              LD      A,(DE)          ;Compare X and Y bytewise
  00A6D 96              SUB     (HL)
                                                ;Byte from X = Byte from Y ?
  00A6E C2230A          JP      NZ,0A23H        ;No: contineu at 0A23H

  00A71 1B              DEC     DE              ;Pointer on Y - 1
  00A72 2B              DEC     HL              ;Pointer on X - 1
  00A73 05              DEC     B               ;Counter - 1
  00A74 20F6            JR      NZ,0A6CH        ;Compare next byte from X and Y

  00A76 C1              POP     BC              ;Remove RET address (095EH)
  00A77 C9              RET


; CP X , Y (DBL)
;
; I: -
; O: if X < Y then A = FFH, C-flag = 1, Z-flag = 0, S-flag = 1
;    if X = Y then A = 00H, C-flag = 0, Z-flag = 1, S-flag = 0
;    if X > Y then A = 01H, C-flag = 0, Z-flag = 0, S-flag = 0

  00A78 CD4F0A          CALL    0A4FH           ;Compare X and Y
  00A7B C25E09          JP      NZ,095EH        ;If X <> Y then goto TEST2

  00A7E C9              RET


; X = CINT (X)
; ------------

  00A7F E7              RST     20H             ;TSTTYP
  00A80 2A2141          LD      HL,(4121H)      ;HL -> X
                                                ;INT type ?
  00A83 F8              RET     M               ;Yes: return
                                                ;STR type ?
  00A84 CAF60A          JP      Z,0AF6H         ;Yes: ?TM Error
                                                ;DBL type ?
  00A87 D4B90A          CALL    NC,0AB9H        ;Yes: convert to SNG
  00A8A 21B207          LD      HL,07B2H        ;Set new return address
  00A8D E5              PUSH    HL              ;to 07B2H (= ?OV Error)
  00A8E 3A2441          LD      A,(4124H)       ;A = Exp (X)
  00A91 FE90            CP      90H             ;Exp > (2 ^ 16) ? (16 bit)
  00A93 300E            JR      NC,0AA3H        ;Yes: continue at 0AA3H

  00A95 CDFB0A          CALL    0AFBH           ;DE = INT (X)
  00A98 EB              EX      DE,HL           ;HL = INT value
  00A99 D1              POP     DE              ;Remove new return address
```

```
; This routine writes the contents of HL into the X register as INT
;
; I: HL = integer value
; O: X = integer value
;    A = VT of X (INT)

  00A9A 222141          LD      (4121H),HL      ;Write HL in X

; Set VT to INT

  00A9D 3E02            LD      A,02H           ;A = type code INT
  00A9F 32AF40          LD      (40AFH),A       ;Save VT
  00AA2 C9              RET


; Continuation of CINT (X)
; X = -32768 ? (Is it still within the INT format ?)

  00AA3 018090          LD      BC,9080H        ;BCDE  = -32768
  00AA6 110000          LD      DE,0000H
  00AA9 CD0C0A          CALL    0A0CH           ;Compare X and BCDE
                                                ;X = BCDE ?
  00AAC C0              RET     NZ              ;No: ?OV Error

  00AAD 61              LD      H,C             ;HL = -32768
  00AAE 6A              LD      L,D
  00AAF 18E8            JR      0A99H           ;Remove RET address and
                                                ;set X = HL

; X = CSNG (X)
; ------------

  00AB1 E7              RST     20H             ;TSTTYP
                                                ;SNG type ?
  00AB2 E0              RET     PO              ;Yes: return
                                                ;INT type ?
  00AB3 FACC0A          JP      M,0ACCH         ;Yes: continue at 0ACCH
                                                ;STR type ?
  00AB6 CAF60A          JP      Z,0AF6H         ;Yes: ?TM Error
                                                ;DBL type:
  00AB9 CDBF09          CALL    09BFH           ;BCDE = X
  00ABC CDEF0A          CALL    0AEFH           ;VT = SNG
  00ABF 78              LD      A,B             ;BCDE = 0 ?
  00AC0 B7              OR      A
  00AC1 C8              RET     Z               ;Yes: ok

  00AC2 CDDF09          CALL    09DFH           ;Adjust mantissas
  00AC5 212041          LD      HL,4120H
  00AC8 46              LD      B,(HL)          ;B = 3rd LSB (X) with DBL
  00AC9 C39607          JP      0796H           ;X = BCDE and round

; X = CSNG (X) (INT)

  00ACC 2A2141          LD      HL,(4121H)      ;HL = INT value

; X = CSNG (HL) (INT)

  00ACF CDEF0A          CALL    0AEFH           ;VT = SNG
  00AD2 7C              LD      A,H             ;A = MSB (HL)
  00AD3 55              LD      D,L             ;D = LSB (HL)
  00AD4 1E00            LD      E,00H           ;E = 00H
  00AD6 0690            LD      B,90H           ;B = 90H (Exp = 2 ^ 16)
  00AD8 C36909          JP      0969H           ;SFLOAT (X)


; X = CDBL (X)
; ------------

  00ADB E7              RST     20H             ;TSTTYP
                                                ;DBL type ?
  00ADC D0              RET     NC              ;Yes: return
```

```
                                                ;STR type ?
  00ADD CAF60A          JP      Z,0AF6H         ;Yes: ?TM Error
                                                ;INT type ?
  00AE0 FCCC0A          CALL    M,0ACCH         ;Yes: CSGN
  00AE3 210000          LD      HL,0000H
  00AE6 221D41          LD      (411DH),HL      ;Set 4 LSBs to zero
  00AE9 221F41          LD      (411FH),HL

; Set VT to DBL

  00AEC 3E08            LD      A,08H           ;A = type code for DBL
  00AEE 013E04          LD      BC,043EH        ;--

; Set VT to SNG

* 00AEF   3E04          LD      A,04            ;A = type code for SNG
  00AF1 C39F0A          JP      0A9FH           ;VT = A


; Test if X is string type. If not: ?TM Error else return

  00AF4 E7              RST     20H             ;TSTTYP
                                                ;STR type ?
  00AF5 C8              RET     Z               ;Yes: return

  00AF6 1E18            LD      E,18H           ;E = error code for ?TM Error
  00AF8 C3A219          JP      19A2H           ;Continue at error routine


; SUB for INT, FIX, CINT
; I: X = SNG value with Exp <= 98H (2 ^ 24)
;    (so with fractional part)
;    A = Exp (X)
; O: DE = INT value of X

  00AFB 47              LD      B,A             ;BCDE = A
  00AFC 4F              LD      C,A
  00AFD 57              LD      D,A
  00AFE 5F              LD      E,A
  00AFF B7              OR      A               ;Zero ?
  00B00 C8              RET     Z               ;Yes: return with DE = 0000H

  00B01 E5              PUSH    HL              ;Save HL
  00B02 CDBF09          CALL    09BFH           ;BCDE = X
  00B05 CDDF09          CALL    09DFH           ;Adjust mantissas
  00B08 AE              XOR     (HL)            ;A, bit 7 = Sign of BCDE or X
  00B09 67              LD      H,A             ;H, bit 7 = Sign (X)
                                                ;X negative ?
  00B0A FC1F0B          CALL    M,0B1FH         ;Yes: round BCDE
  00B0D 3E98            LD      A,98H           ;A = Exp (2 ^ 24)
                                                ;(24 bit mantissa, no fraction)
  00B0F 90              SUB     B               ;A = 98H - Exp (X)
  00B10 CDD707          CALL    07D7H           ;Shift CDE to the right for
                                                ;A bits (B becomes LSB)
                                                ;Shift out franctional part
  00B13 7C              LD      A,H             ;A, bit 7 = Sign (X)
  00B14 17              RLA                     ;C-flag = sign
                                                ;X negative ?
  00B15 DCA807          CALL    C,07A8H         ;Yes: round up X
  00B18 0600            LD      B,00H           ;Clear LSB
                                                ;X negative ?
  00B1A DCC307          CALL    C,07C3H         ;Yes: Invert CDEB
  00B1D E1              POP     HL              ;Restore HL
  00B1E C9              RET

; Round down BCDE

  00B1F 1B              DEC     DE              ;Round LSBs
  00B20 7A              LD      A,D             ;Test LSBs
  00B21 A3              AND     E               ;Was DE = 0000H ?
  00B22 3C              INC     A               ;Yes: then A now is 00H
```

```
  00B23 C0              RET     NZ              ;Done when LSBs <> 0

  00B24 0B              DEC     BC              ;Round MSBs
  00B25 C9              RET


; X = FIX (X)
; -----------

  00B26 E7              RST     20H             ;TSTTYP
                                                ;INT type ?
  00B27 F8              RET     M               ;Yes: return

  00B28 CD5509          CALL    0955H           ;TEST2
                                                ;X < 0 ?
  00B2B F2370B          JP      P,0B37H         ;No: X = INT(X)

  00B2E CD8209          CALL    0982H           ;X = -X
  00B31 CD370B          CALL    0B37H           ;X = INT(X)
  00B34 C37B09          JP      097BH           ;X = ABS (X) and RET


; X = INT (X)
; -----------

  00B37 E7              RST     20H             ;TSTTYP
                                                ;INT type ?
  00B38 F8              RET     M               ;Yes: return
                                                ;DBL type ?
  00B39 301E            JR      NC,0B59H        ;Yes: continue at 0B59H
                                                ;STR type ?
  00B3B 28B9            JR      Z,0AF6H         ;Yes: ?TM Error


; X = INT (X) (SNG)
;
; I: X = single precision value
; O: X = single precision value without fractional part
;    A = LSB of integer value of X

  00B3D CD8E0A          CALL    0A8EH           ;X = CINT (X) (SNG)
  00B40 212441          LD      HL,4124H        ;HL -> Exp (X)
  00B43 7E              LD      A,(HL)          ;A = Exp (X)
  00B44 FE98            CP      98H             ;Exp >= (2 ^ 24) ?
                                                ;Yes: X has no fraction
  00B46 3A2141          LD      A,(4121H)       ;A = LSB of X
  00B49 D0              RET     NC              ;Done when Exp (X) >= 98H

  00B4A 7E              LD      A,(HL)          ;A = Exp (X)
  00B4B CDFB0A          CALL    0AFBH           ;Remove fractional part
  00B4E 3698            LD      (HL),98H        ;Set Exp to 2 ^ 24
  00B50 7B              LD      A,E             ;A = LSB
  00B51 F5              PUSH    AF              ;Save LSB
  00B52 79              LD      A,C             ;A = MSB
  00B53 17              RLA                     ;Sign to C-flag
  00B54 CD6207          CALL    0762H           ;SFLOAT conversion CDEB to X
  00B57 F1              POP     AF              ;LSB back to A
  00B58 C9              RET


; X = INT (X) (DBL)

  00B59 212441          LD      HL,4124H        ;HL -> Exp (X)
  00B5C 7E              LD      A,(HL)          ;A = Exp (X)
  00B5D FE90            CP      90H             ;Exp (X) < (2 ^ 16)
  00B5F DA7F0A          JP      C,0A7FH         ;Yes: execute CINT
                                                ;Exp (X) = (2 ^ 16)
  00B62 2014            JR      NZ,0B78H        ;No: continue at 0B78H

  00B64 4F              LD      C,A             ;C = 90H (Exp = 2 ^ 16!)
  00B65 2B              DEC     HL              ;HL = MSB (X)
```

```
  00B66 7E              LD      A,(HL)          ;A = MSB (X)
  00B67 EE80            XOR     80H             ;A = 00H, if X = -32768
  00B69 0606            LD      B,06H           ;Test remaining 6 bytes
  00B6B 2B              DEC     HL              ;Pointer - 1
  00B6C B6              OR      (HL)            ;Test byte
  00B6D 05              DEC     B               ;Counter - 1
  00B6E 20FB            JR      NZ,0B6BH        ;Next byte

  00B70 B7              OR      A               ;All bytes = 00H ?
  00B71 210080          LD      HL,8000H        ;HL = -32768 (INT value)
  00B74 CA9A0A          JP      Z,0A9AH         ;Yes: Write HL to X as INT

  00B77 79              LD      A,C             ;A = Exp (X) = 90H (because
                                                ;of 0B64H)
  00B78 FEB8            CP      0B8H            ;Exp >= (2 ^ 56) ?
                                                ;(56 bit mantissa)
  00B7A D0              RET     NC              ;Yes: no fractional part

  00B7B F5              PUSH    AF              ;Save Exp
  00B7C CDBF09          CALL    09BFH           ;BCDE = X (SNG)
  00B7F CDDF09          CALL    09DFH           ;Adjust mantissas
  00B82 AE              XOR     (HL)            ;A, bit 7 = Sign of X or BCDE
  00B83 2B              DEC     HL              ;HL -> Exp (X)
  00B84 36B8            LD      (HL),0B8H       ;Set Exp to 2 ^ 56
  00B86 F5              PUSH    AF              ;Save sign
                                                ;X negative ?
  00B87 FCA00B          CALL    M,0BA0H         ;Yes: round X
  00B8A 212341          LD      HL,4123H        ;HL -> MSB (X)
  00B8D 3EB8            LD      A,0B8H          ;A = Exp for 2 ^ 56
  00B8F 90              SUB     B               ;Compute Exp-difference
  00B90 CD690D          CALL    0D69H           ;(HL) - (HL-7) (corresponds
                                                ;with X) and shift A bits to
                                                ;the right: shift out the
                                                ;fractional part
  00B93 F1              POP     AF              ;Restore sign
                                                ;X negative ?
  00B94 FC200D          CALL    M,0D20H         ;Yes: round up mantissa of X
  00B97 AF              XOR     A               ;A = 00H
  00B98 321C41          LD      (411CH),A       ;Clear LSB-underflow of X
  00B9B F1              POP     AF              ;Restore Exp in A
  00B9C D0              RET     NC              ;Done when CALLed from 12B8H
                                                ;(Otherwise C-flag = 1 because
                                                ;of 0B7AH)
  00B9D C3D80C          JP      0CD8H           ;Continue at DFLOAT


; Round down mantissa of X (DBL)

  00BA0 211D41          LD      HL,411DH        ;HL -> X
  00BA3 7E              LD      A,(HL)          ;A = byte of X
  00BA4 35              DEC     (HL)            ;Round down
  00BA5 B7              OR      A               ;Was the byte 00H ?
  00BA6 23              INC     HL              ;Pointer + 1
  00BA7 28FA            JR      Z,0BA3H         ;Yes: next byte

  00BA9 C9              RET


; SUB for array size calculation with DIM
; DE = DE * BC

  00BAA E5              PUSH    HL              ;Save PTP
  00BAB 210000          LD      HL,0000H        ;Result = 0000H
  00BAE 78              LD      A,B             ;BC = 0 ?
  00BAF B1              OR      C
  00BB0 2812            JR      Z,0BC4H         ;Yes: return with result = 0

  00BB2 3E10            LD      A,10H           ;A = counter for 16 bit
  00BB4 29              ADD     HL,HL           ;Shift result to the left
                                                ;Overflow ?
  00BB5 DA3D27          JP      C,273DH         ;Yes: ?BS Error
```

```
  00BB8 EB                EX      DE,HL           ;Save HL
  00BB9 29                ADD     HL,HL           ;Highest bit of DE to C-flag
  00BBA EB                EX      DE,HL           ;Restore HL
  00BBB 3004              JR      NC,0BC1H        ;Jump when bit = 0

  00BBD 09                ADD     HL,BC           ;Add BC to result
                                                  ;Overflow ?
  00BBE DA3D27            JP      C,273DH         ;Yes: ?BS Error

  00BC1 3D                DEC     A               ;Counter - 1
  00BC2 20F0              JR      NZ,0BB4H        ;Next bit

  00BC4 EB                EX      DE,HL           ;Result in DE
  00BC5 E1                POP     HL              ;Restore PTP
  00BC6 C9                RET


; ISUB: X = HL = DE - HL (INT)
;
; I: DE = Original value
;    HL = Subtractor
; O: HL = Difference when in INT range
;    X  = difference (automatic conversion to SNG format when result not
;    in INT range)

  00BC7 7C                LD      A,H             ;A = MSB of HL
  00BC8 17                RLA                     ;C-flag - sign of HL
  00BC9 9F                SBC     A,A             ;If HL >= 0 then A = 00H
                                                  ;           else A = FFH
  00BCA 47                LD      B,A             ;B = A
  00BCB CD510C            CALL    0C51H           ;HL = = HL, C-flag = 1
  00BCE 79                LD      A,C             ;A = 00H (C = 00H by 0C51H)
  00BCF 98                SBC     A,B             ;if HL was >= 0 then A = FFH
                                                                  else A = 00H
  00BD0 1803              JR      0BD5H           ;Continue at IADD
                                                  ;(do a DE = DE + (-HL)

; IADD: X = HL = DE + HL (INT)
;
; I: DE = 1st sum argument
;    HL = 2nd sum argument
; O: HL = sum when in INT range
;    X  = sum (automatic conversion to SNG format when result not
;    in INT range)

  00BD2 7C                LD      A,H             ;Calculate sign-flag
  00BD3 17                RLA                     ;(See ISUB)
  00BD4 9F                SBC     A,A
  00BD5 47                LD      B,A             ;If HL >= 0 then B = 00H
                                                  ;           else B = FFH
  00BD6 E5                PUSH    HL              ;Save 2nd sum argument
  00BD7 7A                LD      A,D             ;Calculate sign of 2nd arg.
  00BD8 17                RLA
  00BD9 9F                SBC     A,A
  00BDA 19                ADD     HL,DE           ;Perform addition
  00BDB 88                ADC     A,B             ;Process sign-flags and
  00BDC 0F                RRCA                    ;overflow
  00BDD AC                XOR     H               ;A, bit 7 is set, when an
                                                  ;overflow occurs in case of
                                                  ;both are equal or when no
                                                  ;overflow occurs in case of
                                                  ;both signs are not equal
                                                  ;A, bit 7 = 0 ?
  00BDE F2990A            JP      P,0A99H         ;Yes: result is ok
                                                  ;No:
  00BE1 C5                PUSH    BC              ;Save sign flag of 2nd arg.
  00BE2 EB                EX      DE,HL           ;HL = 1st sum argument
  00BE3 CDCF0A            CALL    0ACFH           ;X = CSNG (HL)
  00BE6 F1                POP     AF              ;Restore sign flag
  00BE7 E1                POP     HL              ;Restore 2nd sum argument
```

```
  00BE8 CDA409          CALL    09A4H           ;(SP) = X = 1st argument
  00BEB EB              EX      DE,HL           ;DE = 2nd argument
  00BEC CD6B0C          CALL    0C6BH           ;X = SFLOAT (DE)
  00BEF C38F0F          JP      0F8FH           ;X = X + (SP) (addition in
                                                ;SNG format)

; IMUL: X = HL = DE * HL (INT)
;
; I: DE = multiplicant
;    HL = multiplicator (both values in INT format)
; O: HL = product when in INT range
;    X  = product (automatic conversion to SNG format when result not
;    in INT range)

  00BF2 7C              LD      A,H             ;HL = 0000H ?
  00BF3 B5              OR      L
  00BF4 CA9A0A          JP      Z,0A9AH         ;Yes: Result = 0

  00BF7 E5              PUSH    HL              ;Save multiplicator
  00BF8 D5              PUSH    DE              ;Save multiplicant
  00BF9 CD450C          CALL    0C45H           ;Clear sign, make both numbers
                                                ;positive
  00BFC C5              PUSH    BC              ;Save sign flag (B, bit 7 = 0
                                                ;in case both signs are equal)
  00BFD 44              LD      B,H             ;BC = HL
  00BFE 4D              LD      C,L
  00BFF 210000          LD      HL,0000H        ;Result = 0
  00C02 3E10            LD      A,10H           ;A = counter for 16 bits
  00C04 29              ADD     HL,HL           ;Shift result to the left
                                                ;Overflow ?
  00C05 381F            JR      C,0C26H         ;Yes: continue at 0C26H

  00C07 EB              EX      DE,HL           ;Next bit form De to C-flag
  00C08 29              ADD     HL,HL
  00C09 EB              EX      DE,HL
  00C0A 3004            JR      NC,0C10H        ;Jump if bit not set

  00C0C 09              ADD     HL,BC           ;Add BC to result if bit set
                                                ;Overflow ?
  00C0D DA260C          JP      C,0C26H         ;Yes: continue at 0C26H

  00C10 3D              DEC     A               ;Counter - 1
  00C11 20F1            JR      NZ,0C04H        ;Next bit

  00C13 C1              POP     BC              ;Restore sign flag
  00C14 D1              POP     DE              ;Restore multiplicant
  00C15 7C              LD      A,H             ;Result negative ?
  00C16 B7              OR      A
  00C17 FA1F0C          JP      M,0C1FH         ;Yes: continue at 0C1FH

  00C1A D1              POP     DE              ;Restore multiplicator
  00C1B 78              LD      A,B             ;A = sign-flag
  00C1C C34D0C          JP      0C4DH           ;Set sign of result


; Overflow into bit 15 (sign bit)
; Because both factors were made positive, the result must also be positive.
; This means that the 16th bit must be 0.

  00C1F EE80            XOR     80H             ;A = 00H when H = 80H
  00C21 B5              OR      L               ;HL = 8000 = 32768 ?
                                                ;(no sign)
  00C22 2813            JR      Z,0C37H         ;Yes: continue at 0C37H
                                                ;No:
  00C24 EB              EX      DE,HL           ;HL = multiplicant
  00C25 01C1E1          LD      BC,0E1C1H       ;--

; Overflow at IMUL
; Convert both factors to SNG format and then perform a SMUL

* 00C26   C1            POP     BC              ;B = sign-flag
```

```
* 00C27     E1           POP     HL               ;HL = Multiplicant
  00C28 CDCF0A           CALL    0ACFH            ;X = CSNG (HL)
  00C2B E1               POP     HL               ;HL = Multiplicator
  00C2C CDA409           CALL    09A4H            ;(SP) = X
  00C2F CDCF0A           CALL    0ACFH            ;X = CSNG (HL)
  00C32 C1               POP     BC               ;BCDE = (SP)
  00C33 D1               POP     DE
  00C34 C34708           JP      0847H            ;X = BCDE * X (SNG)

; The result is 32768 (unsigned)

  00C37 78               LD      A,B              ;A = sign-flag
  00C38 B7               OR      A                ;Signs not equal ?
  00C39 C1               POP     BC               ;Correct stack
  00C3A FA9A0A           JP      M,0A9AH          ;Yes: result = 8000H = -32768
                                                  ;No: result = +32768
  00C3D D5               PUSH    DE               ;Save DE
  00C3E CDCF0A           CALL    0ACFH            ;X = CSNG (HL) = -32768
  00C41 D1               POP     DE               ;Restore DE
  00C42 C38209           JP      0982H            ;X = -X (result = +32768)


; Sign test at IMUL:
; Clear sign at DE and HL (make both positive)
;
; I: DE = multiplicant
;    HL = multiplicator
; O: DE = ABS (multiplicator)
;    HL = ABS (multiplicant)
;    If both signs are equal then B, bit 7 = 0
;                            else B, bit 7 = 1

  00C45 7C               LD      A,H              ;Sign from HL
  00C46 AA               XOR     D                ;XORed with sign from D
  00C47 47               LD      B,A              ;Result in B, bit 7
  00C48 CD4C0C           CALL    0C4CH            ;HL = ABS (HL)
  00C4B EB               EX      DE,HL            ;Same with DE
  00C4C 7C               LD      A,H              ;HL >= 0 ?
  00C4D B7               OR      A
  00C4E F29A0A           JP      P,0A9AH          ;Yes: ok

; X = HL = -HL (INT)
;
; I: HL = INT value
; O: HL = negative INT value
;    X = HL and VT = INT

  00C51 AF               XOR     A                ;A = 00H
  00C52 4F               LD      C,A              ;C = 00H
  00C53 95               SUB     L
  00C54 6F               LD      L,A              ;L = 00H - L
  00C55 79               LD      A,C              ;A = 00H
  00C56 9C               SBC     A,H
  00C57 67               LD      H,A              ;H = 00H - H - C-flag
  00C58 C39A0A           JP      0A9AH            ;Write HL to X as INT


; X = -X (INT)
; Conversion to SNG, when X = -32768. This because +32768 is no longer
; in the INT range.

  00C5B 2A2141           LD      HL,(4121H)       ;HL = X
  00C5E CD510C           CALL    0C51H            ;X = HL = -HL
  00C61 7C               LD      A,H              ;HL used to be -32768 ?
  00C62 EE80             XOR     80H
  00C64 B5               OR      L
  00C65 C0               RET     NZ               ;No : value ok

  00C66 EB               EX      DE,HL            ;DE = value
  00C67 CDEF0A           CALL    0AEFH            ;VT = SNG
  00C6A AF               XOR     A                ;A = 0
```

```
  00C6B 0698              LD      B,98H           ;B = Exp (2 ^ 24)
  00C6D C36909            JP      0969H           ;SFLOAT (X)


; X = X - Y = X + (-Y) (DBL)

  00C70 212D41            LD      HL,412DH        ;HL -> MSB (Y)
  00C73 7E                LD      A,(HL)          ;A = MSB (Y)
  00C74 EE80              XOR     80H             ;Y = -Y
  00C76 77                LD      (HL),A          ;Store MSB (Y)

; X = X + Y (DBL)

  00C77 212E41            LD      HL,412EH        ;HL -> Exp (Y)
  00C7A 7E                LD      A,(HL)          ;A = Exp (Y)
  00C7B B7                OR      A               ;Exp (Y) = 0 ?
  00C7C C8                RET     Z               ;Yes: X is the result

  00C7D 47                LD      B,A             ;B = Exp (Y)
  00C7E 2B                DEC     HL              HL -> MSB (Y)
  00C7F 4E                LD      C,(HL)          ;C = MSB (Y)
  00C80 112441            LD      DE,4124H        ;DE -> Exp (X)
  00C83 1A                LD      A,(DE)          ;A = Exp (X)
  00C84 B7                OR      A               ;X = 0 ? (Exp (X) = 0)
  00C85 CAF409            JP      Z,09F4H         ;Yes: X = Y

  00C88 90                SUB     B               ;A = Exp (X) - Exp (Y)
                                                  ;Exp (X) >= Exp (Y)
  00C89 3016              JR      NC,0CA1H        ;Yes: continue at 0CA1H
                                                  ;No: swap X and Y
  00C8B 2F                CPL                     ;A = -A
  00C8C 3C                INC     A               ;A + 1 for 2 complement
  00C8D F5                PUSH    AF              ;Save Exp-difference
  00C8E 0E08              LD      C,08H           ;Swap 8 bytes
  00C90 23                INC     HL              ;HL -> Exp (Y)
  00C91 E5                PUSH    HL              ;Save pointer
  00C92 1A                LD      A,(DE)          ;Swap bytes from (DE)
  00C93 46                LD      B,(HL)          ;and (HL)
  00C94 77                LD      (HL),A
  00C95 78                LD      A,B
  00C96 12                LD      (DE),A
  00C97 1B                DEC     DE              ;Pointer - 1
  00C98 2B                DEC     HL              ;Pointer - 1
  00C99 0D                DEC     C               ;Counter - 1
  00C9A 20F6              JR      NZ,0C92H        ;Next byte

  00C9C E1                POP     HL              ;Restore pointer on Exp (X)
  00C9D 46                LD      B,(HL)          ;B = Exp (X)
  00C9E 2B                DEC     HL
  00C9F 4E                LD      C,(HL)          ;C = MSB (X)
  00CA0 F1                POP     AF              ;Restore Exp-difference
                                                  ;In X now the larger argument
  00CA1 FE39              CP      39H             ;Exp-diff. > (2 ^ 56)
  00CA3 D0                RET     NC              ;Yes: Y is too small, the sum
                                                  ;would not alter X
  00CA4 F5                PUSH    AF              ;Save Exp-difference
  00CA5 CDDF09            CALL    09DFH           ;Adjust mantissas
  00CA8 23                INC     HL              ;HL -> Exp (X)
  00CA9 3600              LD      (HL),00H        ;Set Exp to 0
  00CAB 47                LD      B,A             ;B = sign-flag
  00CAC F1                POP     AF              ;Restore Exp-difference
  00CAD 212D41            LD      HL,412DH        ;HL -> Y
  00CB0 CD690D            CALL    0D69H           ;Shift (HL) upto (HL-7) A bits
                                                  ;to the right. This results
                                                  ;in the same exponent for
                                                  ;X and Y
  00CB3 3A2641            LD      A,(4126H)       ;Copy underflow
  00CB6 321C41            LD      (411CH),A       ;to X
  00CB9 78                LD      A,B             ;A = sign-flag
  00CBA B7                OR      A               ;Equal signs ?
  00CBB F2CF0C            JP      P,0CCFH         ;No: continue at 0CCFH
```

```
  00CBE CD330D           CALL    0D33H            ;Add mantissas
                                                  ;Overflow ?
  00CC1 D20E0D           JP      NC,0D0EH         ;No: continue at 0D0EH

  00CC4 EB               EX      DE,HL            ;Overflow: HL -> Exp (X)
  00CC5 34               INC     (HL)             ;Exp (X) + 1
                                                  ;Overflow ?
  00CC6 CAB207           JP      Z,07B2H          ;Yes: ?OV Error

  00CC9 CD900D           CALL    0D90H            ;X = X / 2, because the Exp + 1
                                                  ;means an X * 2
  00CCC C30E0D           JP      0D0EH            ;Continue at 0D0EH

; Unequal signs: subtract mantissas

  00CCF CD450D           CALL    0D45H            ;Subtract mantissas
  00CD2 212541           LD      HL,4125H         ;HL -> sign-flag
                                                  ;Onderflow ?
  00CD5 DC570D           CALL    C,0D57H          ;Yes: negate mantissa of X


; DFLOAT (DBL)
; Shift DBL-mantissa to  the left until the highest bit of the mantissa
; is 1 and the exponent is as small as possible

  00CD8 AF               XOR     A                ;A = 00H
  00CD9 47               LD      B,A              ;B = shift-counter
  00CDA 3A2341           LD      A,(4123H)        ;A = MSB (X)
  00CDD B7               OR      A                ;MSB = 0 ?
  00CDE 201E             JR      NZ,0CFEH         ;No: test bits of MSB
                                                  ;Yes: Shift X by 1 byte to the
                                                  ;left
  00CE0 211C41           LD      HL,411CH         ;HL -> underflow (X)
  00CE3 0E08             LD      C,08H            ;C = counter for 8 bytes
  00CE5 56               LD      D,(HL)           ;Get new byte
  00CE6 77               LD      (HL),A           ;Use old byte
  00CE7 7A               LD      A,D              ;old byte = new byte
  00CE8 23               INC     HL               ;Pointer + 1
  00CE9 0D               DEC     C                ;Counter - 1
  00CEA 20F9             JR      NZ,0CE5H         ;Next byte

  00CEC 78               LD      A,B              ;A = shift-counter
  00CED D608             SUB     08H              ;Subtract 8 (for 8 bits)
  00CEF FEC0             CP      0C0H             ;-64 reached ? (already 8
                                                  ;bytes shifted ?)
  00CF1 20E6             JR      NZ,0CD9H         ;No: test MSB again

  00CF3 C37807           JP      0778H            ;Yes: all bytes of X were 00H
                                                  ;Set X = 0
; Continue shifting bitwise

  00CF6 05               DEC     B                ;Shift-counter - 1
  00CF7 211C41           LD      HL,411CH         ;HL -> Underflow (X)
  00CFA CD970D           CALL    0D97H            ;Shift X by 1 bit to the left
  00CFD B7               OR      A                ;A = new MSB, bit 7 = 1 ?
  00CFE F2F60C           JP      P,0CF6H          ;No: shift again

  00D01 78               LD      A,B              ;A = shift-counter
  00D02 B7               OR      A                ;Anything shifted ?
  00D03 2809             JR      Z,0D0EH          ;No: X ready

  00D05 212441           LD      HL,4124H         ;HL -> Exp (X)
  00D08 86               ADD     A,(HL)           ;Subtract number of shifted
  00D09 77               LD      (HL),A           ;bits from exponent
                                                  ;Overflow ?
  00D0A D27807           JP      NC,0778H         ;No: X = 0

  00D0D C8               RET     Z                ;Done when exponent = 0

  00D0E 3A1C41           LD      A,(411CH)        ;A = underflow byte
```

```
  00D11 B7              OR      A               ;A, bit 7 = 1
  00D12 FC200D          CALL    M,0D20H         ;Yes: round up X

  00D15 212541          LD      HL,4125H        ;HL -> sign-flag
  00D18 7E              LD      A,(HL)          ;A = sign-flag
  00D19 E680            AND     80H             ;Mask sign. A, bit 7 = 1 when
                                                ;equal sign
  00D1B 2B              DEC     HL
  00D1C 2B              DEC     HL              ;HL -> MSB (X)
  00D1D AE              XOR     (HL)            ;XOR with MSB (X)
  00D1E 77              LD      (HL),A          ;Save new MSB incl. sign
  00D1F C9              RET


; Round up X (DBL)

  00D20 211D41          LD      HL,411DH        ;HL -> LSB of X (DBL)
  00D23 0607            LD      B,07H           ;B = mantissa size of DBL
  00D25 34              INC     (HL)            ;Increment LSB
                                                ;Overflow ?
  00D26 C0              RET     NZ              ;No: return

  00D27 23              INC     HL              ;Next byte
  00D28 05              DEC     B               ;Counter - 1
                                                ;All mantissa bytes done ?
  00D29 20FA            JR      NZ,0D25H        ;No: increment next mantissa
                                                ;byte
  00D2B 34              INC     (HL)            ;Increment Exp (X)
                                                ;Overflow ?
  00D2C CAB207          JP      Z,07B2H         ;Yes: ?OV Error
  00D2F 2B              DEC     HL              ;HL -> MSB mantissa
  00D30 3680            LD      (HL),80H        ;Set mantissa to negative
  00D32 C9              RET


; Add mantissas of X and Y. Result in X

  00D33 212741          LD      HL,4127H        ;HL -> LSB of Y (DBL)
  00D36 111D41          LD      DE,411DH        ;DE -> LSB of X (DBL)
  00D39 0E07            LD      C,07H           ;C = mantissa size of DBL
  00D3B AF              XOR     A               ;A = 0
  00D3C 1A              LD      A,(DE)          ;A = mantissa byte X
  00D3D 8E              ADC     A,(HL)          ;A = A + mantissa byte Y
                                                ;      + C-flag
  00D3E 12              LD      (DE),A          ;Store result in X
  00D3F 13              INC     DE              ;Pointer to X + 1
  00D40 23              INC     HL              ;Pointer to Y + 1
  00D41 0D              DEC     C               ;Counter - 1
                                                ;All mantissa bytes done ?
  00D42 20F8            JR      NZ,0D3CH        ;No, add next mantissa bytes

  00D44 C9              RET


; Subtract mantissas of X and Y. Result in X

  00D45 212741          LD      HL,4127H        ;HL -> LSB of Y (DBL)
  00D48 111D41          LD      DE,411DH        ;DE -> LSB of X (DBL)
  00D4B 0E07            LD      C,07H           ;C = mantissa size of DBL
  00D4D AF              XOR     A               ;A = 0
  00D4E 1A              LD      A,(DE)          ;A = mantissa byte X
  00D4F 9E              SBC     A,(HL)          ;A = A - mantissa byte Y
                                                ;      - C-flag
  00D50 12              LD      (DE),A          ;Store result in X
  00D51 13              INC     DE              ;Pointer to X + 1
  00D52 23              INC     HL              ;Pointer to Y + 1
  00D53 0D              DEC     C               ;Counter - 1
                                                ;All mantissa bytes done ?
  00D54 20F8            JR      NZ,0D4EH        ;No, add next mantissa bytes

  00D56 C9              RET
```

```
; Negate mantissa of X (incl. underflow) and sign

  00D57 7E            LD     A,(HL)        ;A = sign-flag
  00D58 2F            CPL                  ;Negation
  00D59 77            LD     (HL),A        ;Save sign-flag
  00D5A 211C41        LD     HL,411CH      ;HL -> underflow (X)
  00D5D 0608          LD     B,08H         ;Negate 9 bytes
  00D5F AF            XOR    A             ;A = 00H
  00D60 4F            LD     C,A           ;C = 00H

  00D61 79            LD     A,C           ;A = C
  00D62 9E            SBC    A,(HL)        ;A = A - (HL) - C-flag
  00D63 77            LD     (HL),A        ;Save difference
  00D64 23            INC    HL            ;Pointer + 1
  00D65 05            DEC    B             ;Counter - 1
  00D66 20F9          JR     NZ,0D61H      ;Next byte

  00D68 C9            RET


; Shift (HL) upto (HL-7) by A bits to the right

  00D69 71            LD     (HL),C        ;Save MSB
  00D6A E5            PUSH   HL            ;Save pointer
  00D6B D608          SUB    08H           ;More then 8 bits to shift ?
  00D6D 380E          JR     C,0D7DH       ;No: shift bits at 0D7DH

  00D6F E1            POP    HL            ;Restore pointer
  00D70 E5            PUSH   HL            ;And save it again
  00D71 110008        LD     DE,0800H      ;Fill 8 bytes mantissa with 00H
  00D74 4E            LD     C,(HL)        ;C = new byte
  00D75 73            LD     (HL),E        ;(HL) = old byte
  00D76 59            LD     E,C           ;Old byte = new byte
  00D77 2B            DEC    HL            ;Pointer - 1
  00D78 15            DEC    D             ;Counter - 1
  00D79 20F9          JR     NZ,0D74H      ;Shift next byte

  00D7B 18EE          JR     0D6BH         ;Shift anything more ?


; Shift bitwise

  00D7D C609          ADD    A,09H         ;Reverse the SUB 08H
  00D7F 57            LD     D,A           ;D = counter
  00D80 AF            XOR    A             ;A = 00H
  00D81 E1            POP    HL            ;Restore pointer
  00D82 15            DEC    D             ;Counter - 1
  00D83 C8            RET    Z             ;RET when done

  00D84 E5            PUSH   HL            ;Save pointer
  00D85 1E08          LD     E,08H         ;8 bytes mantissa
                                           ;(with underflow)
  00D87 7E            LD     A,(HL)        ;A = byte
  00D88 1F            RRA                  ;shift right (incl. C-flag)
  00D89 77            LD     (HL),A        ;Save byte
  00D8A 2B            DEC    HL            ;Pointer - 1
  00D8B 1D            DEC    E             ;Counter - 1
  00D8C 20F9          JR     NZ,0D87H      ;Next byte

  00D8E 18F0          JR     0D80H         ;Any more to shift ?

; Shift mantissa of X one bit to the right

  00D90 212341        LD     HL,4123H      ;HL -> mantissa
  00D93 1601          LD     D,01H         ;D = counter for 1 bit
  00D95 18ED          JR     0D84H         ;Continue at 0D84H

; Shift (HL) to (HL+7) one bit to the right
```

```
  00D97 0E08            LD      C,08H           ;8 bits Mantissa (with
                                                ;underflow)
  00D99 7E              LD      A,(HL)          ;A = byte
  00D9A 17              RLA                     ;shift incl. carry
  00D9B 77              LD      (HL),A          ;Store byte
  00D9C 23              INC     HL              ;Pointer + 1
  00D9D 0D              DEC     C               ;Counter - 1
  00D9E 20F9            JR      NZ,0D99H        ;Shift next byte

  00DA0 C9              RET


; DMUL: X = X * Y (DBL)
;
; I: X = 1st factor (DBL)
;    Y = 2nd factor (DBL)
; O: X = product (DBL)

  00DA1 CD5509          CALL    0955H           ;TEST2
                                                ;X = 0 ?
  00DA4 C8              RET     Z               ;Yes: return

  00DA5 CD0A09          CALL    090AH           ;process exponent and sign
  00DA8 CD390E          CALL    0E39H           ;Save mantissa of 1st factor in
                                                ;414AH and clear mantissa of X
  00DAB 71              LD      (HL),C          ;Underflow of X = 0
  00DAC 13              INC     DE              ;DE -> LSB of 1st factor
  00DAD 0607            LD      B,07H           ;Process 7 bytes mantissa
  00DAF 1A              LD      A,(DE)          ;A = byte from mantissa
  00DB0 13              INC     DE              ;Pointer + 1
  00DB1 B7              OR      A               ;No bit set ?
  00DB2 D5              PUSH    DE              ;Save pointer
  00DB3 2817            JR      Z,0DCCH         ;Yes: shift X 1 byte to the
                                                ;right and get next byte from
                                                ;1st factor
  00DB5 0E08            LD      C,08H           ;8 bits per byte
  00DB7 C5              PUSH    BC              ;Save counter
  00DB8 1F              RRA                     ;Test next bit
                                                ;Bit = 1 ?
  00DB9 47              LD      B,A             ;Save byte first
  00DBA DC330D          CALL    C,0D33H         ;Yes: add mantissa of X and Y
  00DBD CD900D          CALL    0D90H           ;Shift mantissa of X one bit
                                                ;to the right (next position)
  00DC0 78              LD      A,B             ;Restore byte
  00DC1 C1              POP     BC              ;Restore counter
  00DC2 0D              DEC     C               ;Bit counter - 1
  00DC3 20F2            JR      NZ,0DB7H        ;Test next bit

  00DC5 D1              POP     DE              ;Restore pointer
  00DC6 05              DEC     B               ;Byte counter - 1
  00DC7 20E6            JR      NZ,0DAFH        ;Test next byte

  00DC9 C3D80C          JP      0CD8H           ;Continue at DFLOAT

; Shift mantissa of X one byte to the right

  00DCC 212341          LD      HL,4123H        ;HL -> mantissa of X
  00DCF CD700D          CALL    0D70H           ;Shift one byte to the right
  00DD2 18F1            JR      0DC5H           ;Back to DMUL


; Constant 10 (DBL, SNG)

  00DD4 00              DEFB    00H             ;Constant 10 (DBL)
  00DD5 00              DEFB    00H
  00DD6 00              DEFB    00H
  00DD7 00              DEFB    00H
  00DD8 00              DEFB    00H             ;Constant 10 (SNG)
  00DD9 00              DEFB    00H
  00DDA 20              DEFB    20H
  00DDB 84              DEFB    84H
```

```
; X = X / 10 (DBL)

  00DDC 11D40D          LD      DE,0DD4H        ;DE -> Constant 10 (DBL)
  00DDF 212741          LD      HL,4127H        ;HL -> Y
  00DE2 CDD309          CALL    09D3H           ;Copy (DE) to (HL): Y = 10


; DDIV: X = X / Y (DBL)
;
; I: X = dividend (DBL)
;    Y = divisor (DBL)
; O: X = quotient

  00DE5 3A2E41          LD      A,(412EH)       ;A = Exp (X)
  00DE8 B7              OR      A               ;zero ?
  00DE9 CA9A19          JP      Z,199AH         ;Yes: ?/0 Error

  00DEC CD0709          CALL    0907H           ;Process exponent and sign
  00DEF 34              INC     (HL)            ;Adjust exponent
  00DF0 34              INC     (HL)            ;(see also SDIV)
  00DF1 CD390E          CALL    0E39H           ;Save dividend and set mantissa
                                                ;of X to 0
  00DF4 215141          LD      HL,4151H        ;HL -> underflow of dividend
  00DF7 71              LD      (HL),C          ;Clear underflow byte
  00DF8 41              LD      B,C             ;Unterflow flag = 0
  00DF9 114A41          LD      DE,414AH        ;DE -> LSB of dividend
  00DFC 212741          LD      HL,4127H        ;HL -> LSB of divisor
  00DFF CD4B0D          CALL    0D4BH           ;Subtract mantissas
  00E02 1A              LD      A,(DE)          ;A = undeflow byte
  00E03 99              SBC     A,C             ;A = A - C-flag (C-flag = 0)
  00E04 3F              CCF                     ;Invert C-flag
                                                ;Underflow ?
  00E05 380B            JR      C,0E12H         ;No: continue at 0E12H
                                                ;Yes: reverse last subtract
  00E07 114A41          LD      DE,414AH        ;DE -> LSB of dividend
  00E0A 212741          LD      HL,4127H        ;HL -> LSB of divisor
  00E0D CD390D          CALL    0D39H           ;Add mantissas
  00E10 AF              XOR     A               ;C-flag = 0
  00E11 DA1204          JP      C,0412H         ;--
* 00E12   12            LD      (DE),A          ;No underflow: write back
                                                ;undeflow byte
* 00E13     04          INC     B               ;Underflow flag = 1

  00E14 3A2341          LD      A,(4123H)       ;A = MSB of quotient
  00E17 3C              INC     A
  00E18 3D              DEC     A
  00E19 1F              RRA                     ;Shift bit 7 of A to C-flag
                                                ;for rounding purposes
  00E1A FA110D          JP      M,0D11H         ;Done when highest bit = 1

  00E1D 17              RLA                     ;Shift back bit 7 of A
  00E1E 211D41          LD      HL,411DH        ;HL -> LSB of quotient
  00E21 0E07            LD      C,07H           ;7 bytes mantissa
  00E23 CD990D          CALL    0D99H           ;Shift quotient one bit to the
                                                ;left
  00E26 214A41          LD      HL,414AH        ;HL -> LSB of dividend
  00E29 CD970D          CALL    0D97H           ;Shift dividend one bit to the
                                                ;left
  00E2C 78              LD      A,B             ;A = underflow byte
  00E2D B7              OR      A               ;Underflow occured ?
  00E2E 20C9            JR      NZ,0DF9H        ;No: process next bit
                                                ;Yes:
  00E30 212441          LD      HL,4124H        ;HL -> Exp (quotient)
  00E33 35              DEC     (HL)            ;Quotient = Quotient / 2
                                                ;Exp = 0 ?
  00E34 20C3            JR      NZ,0DF9H        ;No: process next bit
                                                ;Yes:
  00E36 C3B207          JP      07B2H           ;?OV Error
```

```
; SUB for DDIV
; Save mantissa of X to 414AH to 4150H and set as result X to zero

  00E39 79                LD      A,C               ;A = MSB (Y)
  00E3A 322D41            LD      (412DH),A         ;Write back MSB (Y)
  00E3D 2B                DEC     HL                ;HL -> MSB (X)
  00E3E 115041            LD      DE,4150H          ;DE -> temp. memory space
  00E41 010007            LD      BC,0700H          ;Copy 7 bytes, set to 00H
  00E44 7E                LD      A,(HL)            ;A = byte from X
  00E45 12                LD      (DE),A            ;Save it in (DE)
  00E46 71                LD      (HL),C            ;Clear mantissa of X
  00E47 1B                DEC     DE                ;Pointer to temp. - 1
  00E48 2B                DEC     HL                ;Pointer to X - 1
  00E49 05                DEC     B                 ;Counter - 1
  00E4A 20F8              JR      NZ,0E44H          ;Next byte

  00E4C C9                RET


; X = X * 10 (DBL)

  00E4D CDFC09            CALL    09FCH             ;Y = X
  00E50 EB                EX      DE,HL             ;HL + 1 -> Exp (X)
  00E51 2B                DEC     HL                ;HL -> Exp (X)
  00E52 7E                LD      A,(HL)            ;A = Exp (X)
  00E53 B7                OR      A                 ;X = 0 ?
  00E54 C8                RET     Z                 ;Yes: done, return

  00E55 C602              ADD     A,02H             ;A = Exp(X) + 2
                                                    ;Overflow ?
  00E57 DAB207            JP      C,07B2H           ;Yes: ?OV Error

  00E5A 77                LD      (HL),A            ;Store Exp: X = X * 4
  00E5B E5                PUSH    HL                ;Save pointer
  00E5C CD770C            CALL    0C77H             ;X = X + Y (now: X = X * 5)
  00E5F E1                POP     HL                ;Restore pointer
  00E60 34                INC     (HL)              ;Exp + 1: X = X * 2
                                                    ;(results in X = X * 10)
  00E61 C0                RET     NZ                ;Return if no overflow

  00E62 C3B207            JP      07B2H             ;?OV Error


; Convert string to number (DBL)
; (like VAL function)
;
; I: HL -> string
; O: X = number (DBL)

  00E65 CD7807            CALL    0778H             ;X = 0
  00E68 CDEC0A            CALL    0AECH             ;set VT to DBL
  00E6B F6AF              OR      0AFH              ;Set flag <> 0


; Convert string in a number according to type (INT, SNG, DBL)
; (like VAL function)
;
; I: HL -> string
; O: X = number

* 00E6C   AF              XOR     A                 ;Set flag = 0
                                                    ;First, the routine will
                                                    ;attempt to convert to an INT.
                                                    ;In case of overflow, it will
                                                    ;attempt to convert to SNG or
                                                    ;to DBL
  00E6D EB                EX      DE,HL             ;DE -> string
  00E6E 01FF00            LD      BC,00FFH          ;B = 00 (number of positions
                                                    ;after the decimal point)
                                                    ;C = FFH (floating point flag:
```

```
                                                    ;see 0EE4H and 0F29H)
  00E71 60              LD      H,B                 ;HL = 0000H
  00E72 68              LD      L,B                 ;(HL is initial value)
  00E73 CC9A0A          CALL    Z,0A9AH             ;Write HL to X as INT
  00E76 EB              EX      DE,HL               ;HL = pointer, DE = 0000H
  00E77 7E              LD      A,(HL)              ;A = string character
  00E78 FE2D            CP      '-'                 ;Test for negative sign
                                                    ;(Negative: Z-flag = 1)
  00E7A F5              PUSH    AF                  ;Save sign
                                                    ;Negative sign ?
  00E7B CA830E          JP      Z,0E83H             ;Yes: continue at 0E83H

  00E7E FE2B            CP      '+'                 ;Positive sign ?
  00E80 2801            JR      Z,0E83H             ;Yes: continue at 0E83H

  00E82 2B              DEC     HL                  ;Pointer - 1 for RST 10H
  00E83 D7              RST     10H                 ;A = next non-space character
                                                    ;Digit ?
  00E84 DA290F          JP      C,0F29H             ;Yes: continue at 0F29H

  00E87 FE2E            CP      '.'                 ;'.' ?
  00E89 CAE40E          JP      Z,0EE4H              Yes: continue at 0EE4H

  00E8C FE45            CP      'E'                 ;'E' ?
  00E8E 2814            JR      Z,0EA4H             ;Yes: continue at 0EA4H

  00E90 FE25            CP      '%'                 ;'%' ? (INT indicator)
  00E92 CAEE0E          JP      Z,0EEEH             ;Yes: continue at 0EEEH

  00E95 FE23            CP      '#'                 ;'#' ? (DBL indicator)
  00E97 CAF50E          JP      Z,0EF5H             ;Yes: continue at 0EF5H

  00E9A FE21            CP      '!'                 ;'!' ? (SNG indicator)
  00E9C CAF60E          JP      Z,0EF6H             ;Yes: continue at 0EF6H

  00E9F FE44            CP      'D'                 ;'D' ?
  00EA1 2024            JR      NZ,0EC7H            ;No: No digit or special
                                                    ;character recognized:
                                                    ;end of number string reached
  00EA3 B7              OR      A                   ;Yes: Set Z-flag = 0

; 'E' (Z-flag = 1) or 'D' (Z-flag = 0)

  00EA4 CDFB0E          CALL    0EFBH               ;Convert X to SNG (Z-flag = 1)
                                                    ;or DBL (Z-flag = 0)
  00EA7 E5              PUSH    HL                  ;Save pointer
  00EA8 21BD0E          LD      HL,0EBDH            ;Set new RET address to 0EBDH
  00EAB E3              EX      (SP),HL             ;and restore pointer
  00EAC D7              RST     10H                 ;A = next character following
                                                    ;'D' or 'E'
  00EAD 15              DEC     D                   ;D = FFH (-1)
  00EAE FECE            CP      0CEH                ;'-' (BASIC token) ?
  00EB0 C8              RET     Z                   ;Yes: continue at 0EBDH

  00EB1 FE2D            CP      '-'                 ;'-' ?
  00EB3 C8              RET     Z                   ;Yes: continue at 0EBDH

  00EB4 14              INC     D                   ;D = 00H (0)
  00EB5 FECD            CP      0CDH                ;'+' (BASIC token) ?
  00EB7 C8              RET     Z                   ;Yes: continue at 0EBDH

  00EB8 FE2B            CP      '+'                 ;'+' ?
  00EBA C8              RET     Z                   ;Yes: continue at 0EBDH

  00EBB 2B              DEC     HL                  ;Pointer - 1 (for RST 10H)
  00EBC F1              POP     AF                  ;Remove RET address (0EBDH)
  00EBD D7              RST     10H                 ;A = Exponent character
                                                    ;Digit found ?
  00EBE DA940F          JP      C,0F94H             ;yes: continue at 0F94H
                                                    ;No: exponent finished
  00EC1 14              INC     D                   ;Exponent negative ? (D was FFH)
```

```
  00EC2 2003            JR      NZ,0EC7H        ;No: continue at 0EC7H
                                                ;Yes:
  00EC4 AF              XOR     A               ;Negate exponent
  00EC5 93              SUB     E               ;A = 00H - E
  00EC6 5F              LD      E,A             ;E = correct exponent

; Number in X is ready: process exponent / floating point / sign

  00EC7 E5              PUSH    HL              ;Save pointer
  00EC8 7B              LD      A,E             ;A = exponent
  00EC9 90              SUB     B               ;A = difference between
                                                ;exponent and number of
                                                ;positions after the decimal
                                                ;point.
                                                ;Larger than exponent ?
  00ECA F40A0F          CALL    P,0F0AH         ;Yes: multiply X with 10
  00ECD FC180F          CALL    M,0F18H         ;No: divide X by 10 and
                                                ;difference + 1
                                                ;Difference = 0 ?
  00ED0 20F8            JR      NZ,0ECAH        ;No: continue processing

  00ED2 E1              POP     HL              ;Restore pointer
  00ED3 F1              POP     AF              ;Restore sign
  00ED4 E5              PUSH    HL              ;Save pointer
                                                ;Negative sign ?
  00ED5 CC7B09          CALL    Z,097BH         ;Yes: X = -X
  00ED8 E1              POP     HL              ;Restore pointer

  00ED9 E7              RST     20H             ;TSTTYP
                                                ;DBL type ?
  00EDA E8              RET     PE              ;Yes : return

  00EDB E5              PUSH    HL              ;Save pointer
  00EDC 219008          LD      HL,0890H        ;Set RET address to POP HL
  00EDF E5              PUSH    HL              ;(to restore pointer)
  00EE0 CDA30A          CALL    0AA3H           ;X = -32768 ?
                                                ;Yes: convert X to INT
  00EE3 C9              RET                     ;RET and restore pointer

; '.' found

  00EE4 E7              RST     20H             ;TSTTYP
  00EE5 0C              INC     C               ;If point found for the first
                                                ;time then C = 00H else C > 00H
                                                ;Number done (C > 0) ?
  00EE6 20DF            JR      NZ,0EC7H        ;Yes: continue at 0EC7H
                                                ;X in INT format ?
  00EE8 DCFB0E          CALL    C,0EFBH         ;Yes: convert to SNG
  00EEB C3830E          JP      0E83H           ;Get next character

; '%' (INT indicator) found

  00EEE E7              RST     20H             ;TSTTYP
                                                ;INT type ?
  00EEF F29719          JP      P,1997H         ;No: ?SN Error

  00EF2 23              INC     HL              ;Pointer + 1
  00EF3 18D2            JR      0EC7H           ;Number is ready

; '#' (DBL indicator) found

  00EF5 B7              OR      A               ;Z-flag = 0

; '!' (SNG indicator) found

  00EF6 CDFB0E          CALL    0EFBH           ;Convert number to SNG
                                                ;(Z-flag=1) or DBL (Z-flag=0)
  00EF9 18F7            JR      0EF2H           ;Pointer + 1, number is ready


; Conversion into SNG or DBL
```

```
;
; I: Z-flag = 1: X = CSNG ( X )
;    Z-flag = 0: X = CDBL ( X )
; O: -

  00EFB E5               PUSH    HL              ;Save registers
  00EFC D5               PUSH    DE
  00EFD C5               PUSH    BC
  00EFE F5               PUSH    AF
                                                 ;Convert to SNG ?
  00EFF CCB10A           CALL    Z,0AB1H         ;Yes: CSNG

  00F02 F1               POP     AF              ;Restore flags
                                                 ;Convert to DBL ?
  00F03 C4DB0A           CALL    NZ,0ADBH        ;Yes: CDBL

  00F06 C1               POP     BC              ;Restore registers
  00F07 D1               POP     DE
  00F08 E1               POP     HL
  00F09 C9               RET


; X = X * 10 (SNG,DBL)
; Type conform multiplication with 10 of X
; Used at processing of the exponent and fractional part
;
; I: X = number (SNG or DBL format)
; O: X = X * 10
;    A = A - 1 (for exponent and fractional part processing)

                                                 ;Difference between exponent
                                                 ;and fractional part = 0 ?
                                                 ;(Z-flag = 1)
  00F0A C8               RET     Z               ;Yes: done, return

  00F0B F5               PUSH    AF              ;Save difference
  00F0C E7               RST     20H             ;TSTTYP
  00F0D F5               PUSH    AF              ;Save flags
                                                 ;SNG type ?
  00F0E E43E09           CALL    PO,093EH        ;Yes: X = X * 10 (SNG)
  00F11 F1               POP     AF              ;Restore flags
                                                 ;DBL type ?
  00F12 EC4D0E           CALL    PE,0E4DH        ;Yes: X = X * 10 (DBL)
  00F15 F1               POP     AF              ;Restore difference
  00F16 3D               DEC     A               ;Difference - 1
  00F17 C9               RET


; X = X / 10 (SNG,DBL)
; Type conform division by 10 of X
;
; I: X = number (SNG or DBL format)
; O: X = X / 10
;    A = A + 1 (for exponent and fractional part processing)

  00F18 D5               PUSH    DE              ;Save registers
  00F19 E5               PUSH    HL
  00F1A F5               PUSH    AF
  00F1B E7               RST     20H             ;TSTTYP
  00F1C F5               PUSH    AF              ;Save flags
                                                 ;SNG type ?
  00F1D E49708           CALL    PO,0897H        ;Yes: X = X / 10 (SNG)
  00F20 F1               POP     AF              ;Restore flags
                                                 ;DBL type ?
  00F21 ECDC0D           CALL    PE,0DDCH        ;Yes: X = X / 10 (DBL)
  00F24 F1               POP     AF              ;Restore registers
  00F25 E1               POP     HL
  00F26 D1               POP     DE
  00F27 3C               INC     A               ;Difference + 1
  00F28 C9               RET
```

```
; Process digit (C-flag = 1 because of previous RST 10H)

  00F29 D5              PUSH    DE              ;Save exponent flags
  00F2A 78              LD      A,B             ;A = number of fractional
                                                ;positions
  00F2B 89              ADC     A,C             ;C = FFH
                                                ;If no fractional part
                                                ;recognized then
                                                ;C = FFH + C-flag (1), so A
                                                ;remains the same else
                                                ;C = 00H + C-flag (1) = 1, so
                                                ;the number of positions behind
                                                ;the decimal pointer is
                                                ;incremented by one
  00F2C 47              LD      B,A             ;Result in B
  00F2D C5              PUSH    BC              ;Save BC
  00F2E E5              PUSH    HL              ;Save pointer
  00F2F 7E              LD      A,(HL)          ;A = digit (ASCII: '0' to '9')
  00F30 D630            SUB     30H             ;A = value (numerical: 0 to 9)
  00F32 F5              PUSH    AF              ;Save digit
  00F33 E7              RST     20H             ;TSTTYP
                                                ;X still INT type ?
  00F34 F25D0F          JP      P,0F5DH         ;No: continue at 0F5DH

; Insert new digit into INT number

  00F37 2A2141          LD      HL,(4121H)      ;HL -> X (INT number)
  00F3A 11CD0C          LD      DE,0CCDH        ;DE = 3277 (approx. 32767/10)
  00F3D DF              RST     18H             ;Current number already larger
                                                ;than 3277 ?
  00F3E 3019            JR      NC,0F59H        ;Yes, inserting the new digit
                                                ;will cause the number to
                                                ;go outside of the INT range:
                                                ;the number has to be
                                                ;converted to SNG
  00F40 54              LD      D,H             ;DE = number
  00F41 5D              LD      E,L
  00F42 29              ADD     HL,HL           ;HL = HL * 2  = number *  2
  00F43 29              ADD     HL,HL           ;HL = HL * 2  = number *  4
  00F44 19              ADD     HL,DE           ;HL = HL + DE = number *  5
  00F45 29              ADD     HL,HL           ;HL = HL * 2  = number * 10
  00F46 F1              POP     AF              ;Restore digit value
  00F47 4F              LD      C,A             ;BC = digit value
                                                ;(B = fractional part = 0,
                                                ;because the number is INT)
  00F48 09              ADD     HL,BC           ;HL = number + new digit value
  00F49 7C              LD      A,H             ;New number > 32767
  00F4A B7              OR      A
  00F4B FA570F          JP      M,0F57H         ;Yes: convert number to SNG

  00F4E 222141          LD      (4121H),HL      ;X = HL = new number
  00F51 E1              POP     HL              ;Restore pointer
  00F52 C1              POP     BC              ;Restore decimal point
  00F53 D1              POP     DE              ;Restore exponent flag
  00F54 C3830E          JP      0E83H           ;Get next digit

; Overflow of INT

  00F57 79              LD      A,C             ;A = digit value
  00F58 F5              PUSH    AF              ;Save it
  00F59 CDCC0A          CALL    0ACCH           ;X = CSNG ( X )
  00F5C 37              SCF                     ;Set C-flag to 1 for indicate
                                                ;SNG processing

; Insert new digit into SNG (C-flag = 1) or DBL (c-flag = 0) number

                                                ;DBL ?
  00F5D 3018            JR      NC,0F77H        ;Yes: continue at 0F77H

  00F5F 017494          LD      BC,9474H        ;BCDE = 1E+6
```

```
  00F62 110024           LD      DE,2400H
  00F65 CD0C0A           CALL    0A0CH            ;Compare X and BCDE
                                                  ;X already >= 1E+6 ?
  00F68 F2740F           JP      P,0F74H          ;Yes: convert X to DBL, because
                                                  ;the new number X gets more
                                                  ;than 6 positions and will go
                                                  ;outide the SNG range
  00F6B CD3E09           CALL    093EH            ;X = X * 10
  00F6E F1               POP     AF               ;A = digit value
  00F6F CD890F           CALL    0F89H            ;X = X + A (SNG)
  00F72 18DD             JR      0F51H            ;Get next digit

; Overflow of SNG

  00F74 CDE30A           CALL    0AE3H            ;X = CDBL ( X )

; Insert new digit into DBL number

  00F77 CD4D0E           CALL    0E4DH            ;X = X * 10 (DBL)
  00F7A CDFC09           CALL    09FCH            ;Y = X
  00F7D F1               POP     AF               ;A = digit value
  00F7E CD6409           CALL    0964H            ;X = A
  00F81 CDE30A           CALL    0AE3H            ;X = CDBL ( X )
  00F84 CD770C           CALL    0C77H            ;X = X + Y
  00F87 18C8             JR      0F51H            ;Get next digit

; X = X + A (SNG)

  00F89 CDA409           CALL    09A4H            ;(SP) = X
  00F8C CD6409           CALL    0964H            ;X = A

; X = X + (SP) (SNG)

  00F8F C1               POP     BC               ;BCDE = (SP)
  00F90 D1               POP     DE
  00F91 C31607           JP      0716H            ;X = X + BCDE

; Digit found after 'E' or 'D'

  00F94 7B               LD      A,E              ;A = current exponent
  00F95 FE0A             CP      0AH              ;Exponent >= 10 ?
                                                  ;Already two exponent postions
                                                  ;recognized ?
                                                  ;(one position can only give
                                                  ;a maximum of 9)
  00F97 3009             JR      NC,0FA2H         ;Yes: set exponent to 48 to
                                                  ;force an overflow
  00F99 07               RLCA                     ;A = A * 2 = exponent *  2
  00F9A 07               RLCA                     ;A = A * 2 = exponent *  4
  00F9B 83               ADD     A,E              ;A = A * E = exponent *  5
  00F9C 07               RLCA                     ;A = A * 2 = exponent * 10
  00F9D 86               ADD     A,(HL)           ;Insert new exponent digit
  00F9E D630             SUB     30H              ;Subtract 30H, because an
                                                  ;ASCII value was inserted
                                                  ;(Result is always positive,
                                                  ;because (HL) is in the range
                                                  ;from 30H to 39H ('0' to '9'))
  00FA0 5F               LD      E,A              ;E = new exponent
  00FA1 FA1E32           JP      M,321EH          ;-- (positive result!)
* 00FA1   1E32           LD      E,32H            ;Exponent = 48 at overflow
  00FA4 C3BD0E           JP      0EBDH            ;Get next exponent digit


; Print 'in' followed by number in HL (routine for Error and Break)

  00FA7 E5               PUSH    HL               ;Save number
  00FA8 212419           LD      HL,1924H         ;HL -> ' in '
  00FAB CDA728           CALL    28A7H            ;Print text
  00FAE E1               POP     HL               ;Restore number

; Print HL as decimal number (routine for printing line numbers with LIST)
```

```
  00FAF CD9A0A          CALL    0A9AH           ;Write HL to X as INT
  00FB2 AF              XOR     A               ;A = 00H: no formatting
  00FB3 CD3410          CALL    1034H           ;Store formatting byte and
                                                ;clear sign
  00FB6 B6              OR      (HL)            ;A = 20H (bit 7 of A = 0)
  00FB7 CDD90F          CALL    0FD9H           ;Generate unformatted string
  00FBA C3A628          JP      28A6H           ;Print string


; Conversion of X into an unformatted string (for PRINT)
; (like STR$ function)
;
; I: X = number
; O: HL -> string (= 4130H)

  00FBD AF              XOR     A               ;Clear format byte

; Conversion of X into a formatted string (for PRINT USING)
; (like STR$ function)
;
; I: X = number
;    A = formatting code: Bit 7 = 1: do a format
;                         Bit 6 = 1: print ',' for separation of thousands
;                         Bit 5 = 1: fill leading spaces with '*'
;                         Bit 4 = 1: print '$' in front of number
;                         Bit 3 = 1: print sign (also '+')
;                         Bit 2 = 1: print sign behind number
;                         Bit 1 = -: not used
;                         Bit 0 = 1: print exponent
;
; O: HL -> start of string (= 4130H)
;    DE -> end of string

  00FBE CD3410          CALL    1034H           ;Store format byte
                                                ;Clear sign position in buffer
                                                ;and set HL to start of buffer
                                                ;( = 4130H)
  00FC1 E608            AND     08H             ;Sign requested ?
  00FC3 2802            JR      Z,0FC7H         ;No: continue at 0FC7H

  00FC5 362B            LD      (HL),2BH        ;First set sign to '+'
  00FC7 EB              EX      DE,HL           ;Save HL in DE
  00FC8 CD9409          CALL    0994H           ;TEST1
  00FCB EB              EX      DE,HL           ;Restore HL
                                                ;X positive ?
  00FCC F2D90F          JP      P,0FD9H         ;Yes: leave sign as it is

  00FCF 362D            LD      (HL),2DH        ;Set sign to '-'
  00FD1 C5              PUSH    BC              ;Save BC
  00FD2 E5              PUSH    HL              ;Save HL
  00FD3 CD7B09          CALL    097BH           ;X = -X
                                                ;(X is processed as a positive
                                                ;value because the sign is
                                                ;already in the buffer)
  00FD6 E1              POP     HL              ;Restore HL
  00FD7 C1              POP     BC              ;Restore BC
  00FD8 B4              OR      H               ;Z-flag = 0
                                                ;X is now positive
                                                ;Z-flag = 1 in case X = 0
  00FD9 23              INC     HL              ;Pointer + 1
  00FDA 3630            LD      (HL),30H        ;Put '0' in buffer
  00FDC 3AD840          LD      A,(40D8H)       ;A = format byte
  00FDF 57              LD      D,A             ;Save it in D
  00FE0 17              RLA                     ;C-flag = bit 7 of A
  00FE1 3AAF40          LD      A,(40AFH)       ;A = VT
                                                ;Formatting required ?
  00FE4 DA9A10          JP      C,109AH         ;Yes: continue at 109AH
                                                ;Number is zero ?
  00FE7 CA9210          JP      Z,1092H         ;Yes: done, continue at 1092H
```

```
  00FEA FE04            CP      04H             ;Is X an INT (VT < 4) ?
  00FEC D23D10          JP      NC,103DH        ;No: continue at 103DH

; Convert INT number to unformatted string

  00FEF 010000          LD      BC,0000H        ;B = 0: do not generate a
                                                ;decimal point
                                                ;C = 0: no separator of
                                                ;thousands
  00FF2 CD2F13          CALL    132FH           ;Convert number into
                                                ;unformatted string using 5
                                                ;digits (incl. leading zeroes)

; Delete leading zeroes or replace them by '*'

  00FF5 213041          LD      HL,4130H        ;HL -> buffer
  00FF8 46              LD      B,(HL)          ;B = sign (' ' or '-')
  00FF9 0E20            LD      C,' '           ;C = ' '
  00FFB 3AD840          LD      A,(40D8H)       ;A = format byte
  00FFE 5F              LD      E,A             ;E = format byte
  00FFF E620            AND     20H             ;Fill space with '*' ?
  01001 2807            JR      Z,100AH         ;No: continue at 100AH

  01003 78              LD      A,B             ;A = sign
  01004 B9              CP      C               ;Sign = ' ' ?
  01005 0E2A            LD      C,'*'           ;C = '*'
  01007 2001            JR      NZ,100AH        ;No: sign = '-', goto 100AH

  01009 41              LD      B,C             ;Replace sign (' ') by '*'

  0100A 71              LD      (HL),C          ;Write ' ' or '*' into buffer
  0100B D7              RST     10H             ;A = next character
                                                ;End of string ?
  0100C 2814            JR      Z,1022H         ;Yes: continue at 1022H

  0100E FE45            CP      'E'             ;'E' found ?
  01010 2810            JR      Z,1022H         ;Yes: string end ('E' does
                                                ;not belong to INT format)
  01012 FE44            CP      'D'             ;'D' found ?
  01014 280C            JR      Z,1022H         ;Yes: string end ('D' does
                                                ;not belong to INT format)
  01016 FE30            CP      '0'             ;Leading zero ?
  01018 28F0            JR      Z,100AH         ;Yes: replace by ' 'or '*'

  0101A FE2C            CP      ','             ;',' found ?
  0101C 28EC            JR      Z,100AH         ;Yes: replace by ' ' or '*'

  0101E FE2E            CP      '.'             ;'.' found
  01020 2003            JR      NZ,1025H        ;No: continue at 1025H

  01022 2B              DEC     HL              ;Yes: decimal point found
  01023 3630            LD      (HL),30H        ;Replace by '0'
  01025 7B              LD      A,E             ;A = format byte
  01026 E610            AND     10H             ;'$' in front of number ?
  01028 2803            JR      Z,102DH         ;No: continue at 102DH

  0102A 2B              DEC     HL              ;Yes: insert '$'
  0102B 3624            LD      (HL),'$'
  0102D 7B              LD      A,E             ;A = format byte
  0102E E604            AND     04H             ;Print sign behind number ?
  01030 C0              RET     NZ              ;Yes: string is ready, return

  01031 2B              DEC     HL              ;No: put back sign in front of
  01032 70              LD      (HL),B          ;number
  01033 C9              RET

; Save format byte, set HL to start of buffer and clear sign

  01034 32D840          LD      (40D8H),A       ;Save format byte in system RAM
  01037 213041          LD      HL,4130H        ;HL -> start of buffer
  0103A 3620            LD      (HL),' '        ;Clear sign in buffer
```

```
  0103C C9              RET

; X is in floating point format
; Generate unformatted string

  0103D FE05            CP      05H             ;if SNG then C-flag = 1
                                                ;       else C-flag = 1
  0103F E5              PUSH    HL              ;Save pointer
  01040 DE00            SBC     A,00H           ;If SNG then A = 3
                                                ;if DBL then A = 8
  01042 17              RLA                     ;* 2 gives the number of
                                                ;decimal positions to be
                                                ;generated  - 1
  01043 57              LD      D,A             ;D = A
  01044 14              INC     D               ;D = maximum number of
                                                ;positions (7 for SNG,
                                                ;17 for DBL)
  01045 CD0112          CALL    1201H           ;Scale X to 6 / 16 positions
                                                ;A = exponent offset
                                                ;( = number of decimal point
                                                ;shifts to the left during
                                                ;scaling)
  01048 010003          LD      BC,0300H        ;B = 3 (decimal point pos. + 1)
                                                ;C = 0 (no thousands separator)
  0104B 82              ADD     A,D             ;A = exponent-offset + maximum
                                                ;number of positions =
                                                ;10-exponent + 2
                                                ;10-exponent < -2 ?
  0104C FA5710          JP      M,1057H         ;Yes: continue at 1057H

  0104F 14              INC     D               ;Position mnumber + 1 smaller
  01050 BA              CP      D               ;than 10-exponent + 2
  01051 3004            JR      NC,1057H        ;Yes: continue at 1057H

  01053 3C              INC     A               ;10-exponent + 3 =
                                                ;decimal point position + 1
  01054 47              LD      B,A             ;B = decimal point pos. + 1
  01055 3E02            LD      A,02H           ;A = 2 because of SUB 02H:
                                                ;no 10-exponent is printed
  01057 D602            SUB     02H             ;A = 10-exponent
  01059 E1              POP     HL              ;Restore buffer pointer
  0105A F5              PUSH    AF              ;Save 10-exponent
  0105B CD9112          CALL    1291H           ;Set ',' and '.', B - 1
  0105E 3630            LD      (HL),'0'        ;Use '0'
  01060 CCC909          CALL    Z,09C9H         ;Pointer + 1 when decimal point
                                                ;set
  01063 CDA412          CALL    12A4H           ;Convert X into unformatted
                                                ;string with 7 / 17 positions
                                                ;Use decimal point after B
                                                ;positions
  01066 2B              DEC     HL              ;Pointer - 1
  01067 7E              LD      A,(HL)          ;A = character
  01068 FE30            CP      '0'             ;Trailing zero ?
  0106A 28FA            JR      Z,1066H         ;Yes: pointer = last character
                                                ;unequal to '0'. (trailing
                                                ;zeroes are represented by
                                                ;10-exponent)
  0106C FE2E            CP      '.'             ;Last character is the decimal
                                                ;point ?
                                                ;Yes: leave pointer alone,
                                                ;decimal point is deleted
  0106E C4C909          CALL    NZ,09C9H        ;No: pointer + 1
  01071 F1              POP     AF              ;Restore 10-exponent
                                                ;10-exponent = 0 ?
  01072 281F            JR      Z,1093H         ;Yes: done, continue at 1093H

; Use 10-exponent
; A = 10-exponent

  01074 F5              PUSH    AF              ;Save 10-exponent
  01075 E7              RST     20H             ;TSTTYP
```

```
  01076 3E22              LD      A,22H           ;A = ASCII value of 'D' / 2
  01078 8F                ADC     A,A             ;A = A *2 + C-flag
                                                  ;A = 'E' (X in SNG format)
                                                  ;A = 'D' (X in DBL format)
  01079 77                LD      (HL),A          ;Put correct exponent character
                                                  ;in buffer
  0107A 23                INC     HL              ;Pointer + 1
  0107B F1                POP     AF              ;Restore 10-exponent
  0107C 362B              LD      (HL),'+'        ;Assume positve sign
                                                  ;10-exponent > 0 ?
  0107E F28510            JP      P,1085H         ;Yes: continue at 1085H

  01081 362D              LD      (HL),'-'        ;Use '-' instead
  01083 2F                CPL                     ;Negate exponent (so treat it
  01084 3C                INC     A               ;as a positive number)
  01085 062F              LD      B,2FH           ;B = ASCII value of '0' - 1
  01087 04                INC     B               ;B + 1 (next digit in the
                                                  ;tenth position
  01088 D60A              SUB     0AH             ;Subtract 10 for 10-exponent
                                                  ;10-exponent < 10 ?
  0108A 30FB              JR      NC,1087H        ;No: increment digit again

  0108C C63A              ADD     A,3AH           ;+ 3AH gives correct ASCII
                                                  ;value for remaining position
  0108E 23                INC     HL              ;Pointer + 1
  0108F 70                LD      (HL),B          ;Set first position of exp.
  01090 23                INC     HL              ;Pointer + 1
  01091 77                LD      (HL),A          ;Set second position of exp.

; Terminate string with 00H

  01092 23                INC     HL              ;Pointer + 1
  01093 3600              LD      (HL),00H        ;Terminate with 00H
  01095 EB                EX      DE,HL           ;DE -> end of string
  01096 213041            LD      HL,4130H        ;HL -> start of string
  01099 C9                RET


; Formatting required
;
; I: A= VT
;    BC = position counter before and after decimal point
;    D = format byte
;    HL -> buffer

  0109A 23                INC     HL              ;Pointer + 1
  0109B C5                PUSH    BC              ;Save counters
  0109C FE04              CP      04H             ;X in INT format ?
  0109E 7A                LD      A,D             ;A = format byte
  0109F D20911            JP      NC,1109H        ;No: continue at 1109H

; Convert INT number into formatted string

  010A2 1F                RRA                     ;C-flag = bit 0 of A
                                                  ;Use 10-exponent ?
  010A3 DAA311            JP      C,11A3H         ;Yes: continue at 11A3H
                                                  ;(convert number to SNG)
  010A6 010306            LD      BC,0603H        ;B = maximum number of
                                                  ;positions before decimal point
                                                  ;C = counter for thousands
                                                  ;separation
  010A9 CD8912            CALL    1289H           ;Set C = 0 if no thousands
                                                  ;separation required
  010AC D1                POP     DE              ;Restore counters
                                                  ;D = number of positions
                                                  ;before decimal point
                                                  ;E = number of positions
                                                  ;after decimal point
  010AD 7A                LD      A,D             ;A = number of positions
                                                  ;before decimal point
  010AE D605              SUB     05H             ;More then 4 positions ?
```

```
  010B0 F46912          CALL    P,1269H         ;Yes: use corresponding number
                                                ;of leading zeroes
  010B3 CD2F13          CALL    132FH           ;Convert X into 5 digit string
  010B6 7B              LD      A,E             ;E = number of positions
                                                ;after decimal point
  010B7 B7              OR      A               ;Any positions at all ?
  010B8 CC2F09          CALL    Z,092FH         ;Yes: buffer pointer -1
  010BB 3D              DEC     A               ;Any positions required ?
  010BC F46912          CALL    P,1269H         ;Yes: use corresponding number
                                                ;of zeroes
  010BF E5              PUSH    HL              ;Save pointer on end of buffer
  010C0 CDF50F          CALL    0FF5H           ;Delete leading zeroes or
                                                ;replace them by '*'
  010C3 E1              POP     HL              ;Restore buffer pointer
                                                ;Sign in front ?
  010C4 2802            JR      Z,10C8H         ;yes: continue at 10C8H

  010C6 70              LD      (HL),B          ;Set sign behind the number
  010C7 23              INC     HL              ;Pointer + 1
  010C8 3600            LD      (HL),00H        ;Terminate string
  010CA 212F41          LD      HL,412FH        ;HL -> Start of buffer - 1
  010CD 23              INC     HL              ;Pointer + 1
  010CE 3AF340          LD      A,(40F3H)       ;A = LSB of buffer address of
                                                ;decimal point
  010D1 95              SUB     L               ;-LSB of current buffer pointer
  010D2 92              SUB     D               ;Equals the requested number of
                                                ;positions before the decimal
                                                ;point ?
  010D3 C8              RET     Z               ;Yes: done, return

; Shift string in buffer
; Delete a position before the decimal point (means deleting a space)

  010D4 7E              LD      A,(HL)          ;Get next character from buffer
  010D5 FE20            CP      ' '             ;Is it a leading space ?
  010D7 28F4            JR      Z,10CDH         ;Yes: skip it, next character

  010D9 FE2A            CP      '*'             ;'*' in front of number ?
  010DB 28F0            JR      Z,10CDH         ;Yes: skip it, next character

  010DD 2B              DEC     HL              ;Pointer - 1
  010DE E5              PUSH    HL              ;Save pointer
                                                ;(now points on sign character
                                                ;or the first digit or '$')
  010DF F5              PUSH    AF              ;Save character on stack
  010E0 01DF10          LD      BC,10DFH        ;Set new RET address to 10DFH
  010E3 C5              PUSH    BC
  010E4 D7              RST     10H             ;Search start of number string
                                                ;and get next character
  010E5 FE2D            CP      '-'             ;'-' found ?
  010E7 C8              RET     Z               ;Yes: save character, next
                                                ;character
  010E8 FE2B            CP      '+'             ;'+' found ?
  010EA C8              RET     Z               ;Yes: save character, next
                                                ;character
  010EB FE24            CP      '$'             ;'$' found ?
  010ED C8              RET     Z               ;Yes: save character, next
                                                ;character
  010EE C1              POP     BC              ;Remove RET adress
  010EF FE30            CP      '0'             ;Leading zero found ?
  010F1 200F            JR      NZ,1102H        ;No: field overflow!
                                                ;Yes:
  010F3 23              INC     HL              ;Pointer + 1, skip leading zero
  010F4 D7              RST     10H             ;Get next character
                                                ;Is it a digit ?
  010F5 300B            JR      NC,1102H        ;No: field overflow!
                                                ;Yes:
  010F7 2B              DEC     HL              ;Pointer - 1 (string starts
                                                ;one position earlier in
                                                ;the buffer)
  010F8 012B77          LD      BC,772BH        ;--
```

```
* 010F8   2B            DEC     HL              ;Buffer pointer - 1
* 010F8     77          LD      (HL),A          ;Store character back into
                                                ;the buffer
  010FB F1              POP     AF              ;Get character from stack
                                                ;Start of string reached ?
  010FC 28FB            JR      Z,10F9H         ;No: store next character
                                                ;Yes:
  010FE C1              POP     BC              ;Remove buffer pointer from
                                                ;stack
  010FF C3CE10          JP      10CEH           ;Number of positions now ok ?


; Field overflow
; More positions before decimal point are generated as there are reqeusted

  01102 F1              POP     AF              ;Get character from stack
                                                ;Last character ?
  01103 28FD            JR      Z,1102H         ;No: get next character
                                                ;Yes:
  01105 E1              POP     HL              ;Restore buffer pointer to
                                                ;start of string
  01106 3625            LD      (HL),'%'        ;Use '%' to indicate overflow
  01108 C9              RET


; Formatting requested
; X is in floating point format

  01109 E5              PUSH    HL              ;Save pointer
  0110A 1F              RRA                     ;C-flag = bit 0 of A
                                                ;Exponent output requested ?
  0110B DAAA11          JP      C,11AAH         ;Yes: continue at 11AAH
                                                ;X in SNG format ?
  0110E 2814            JR      Z,1124H         ;Yes: continue at 1124H


; X in DBL format
; Generate number string without exponent
; (X must not have more that 16 positions before the decimal point. This
; means that X must be smaller than 1D+16)

  01110 118413          LD      DE,1384H        ;DE -> 1D+16
  01113 CD490A          CALL    0A49H           ;CP X,(DE) = CP X,1D+16
  01116 1610            LD      D,10H           ;D = maximum number of
                                                ;positions (16)
                                                ;X < 1D+16 ?
  01118 FA3211          JP      M,1132H         ;Yes: continue at 1132H

; Field overflow with floating point number

  0111B E1              POP     HL              ;Restore buffer pointer
  0111C C1              POP     BC              ;Restore counters
  0111D CDBD0F          CALL    0FBDH           ;Generate unformatted string
  01120 2B              DEC     HL              ;Buffer pointer - 1
  01121 3625            LD      (HL),'%'        ;Use '%' in front of string
                                                ;to indicate overflow
  01123 C9              RET

; X has SNG format

  01124 010EB6          LD      BC,0B60EH       ;BCDE = 1E+16
  01127 11CA1B          LD      DE,1BCAH
  0112A CD0C0A          CALL    0A0CH           ;CP X,BCDE = CP X,1E+16
                                                ;X > 1E+16 ?
  0112D F21B11          JP      P,111BH         ;Yes: field overflow

  01130 1606            LD      D,06H           ;D = maximum number of
                                                ;positions (6)
  01132 CD5509          CALL    0955H           ;TEST2 (number = 0 ?)
  01135 C40112          CALL    NZ,1201H        ;No: scale number to 6 or 16
                                                ;positions
```

```
  01138 E1              POP     HL              ;Restore pointer
  01139 C1              POP     BC              ;Restore counters on number of
                                                ;positions before and after
                                                ;the decimal point
                                                ;Did scale operation extend ?
  0113A FA5711          JP      M,1157H         ;Yes: continue at 1157H

; The scale operation resulted in truncation (no fractional digits)
; A = exponent offset ( > 0 )

  0113D C5              PUSH    BC              ;Save counters
  0113E 5F              LD      E,A             ;E = exponent offset
  0113F 78              LD      A,B             ;A = number of positions before
                                                ;the decimal point
  01140 92              SUB     D               ;- maximum number of positions
  01141 93              SUB     E               ;- exponent offset
  01142 F46912          CALL    P,1269H         ;Use corresponding number of
                                                ;leading zeroes
  01145 CD7D12          CALL    127DH           ;Compute decimal point position
                                                ;and thousand separation
  01148 CDA412          CALL    12A4H           ;Convert floating point number
                                                ;into unformatted string
  0114B B3              OR      E               ;A = exponent offset
  0114C C47712          CALL    NZ,1277H        ;Set corresponding number
                                                ;of trailing zeroes (because
                                                ;10-exponent is not used)
  0114F B3              OR      E               ;A = exponent offset
  01150 C49112          CALL    NZ,1291H        ;Use decimal point if necessary
  01153 D1              POP     DE              ;Restore positions counters
  01154 C3B610          JP      10B6H           ;Number of positions ok ?

; The scale operation resulted in extension (fractional digits presents)
; A = exponent offset ( < 0 )

  01157 5F              LD      E,A             ;E = exponent offset
  01158 79              LD      A,C             ;A = number of requested
                                                ;positions after the decimal
                                                ;point
  01159 B7              OR      A               ;Any positions requested ?
  0115A C4160F          CALL    NZ,0F16H        ;Yes: A - 1 (because of
                                                ;decimal point)
  0115D 83              ADD     A,E             ;A = number of positions
                                                ;after the decimal point
                                                ;+ exponent offset
                                                ;= negative number of the
                                                ;superfluous generated
                                                ;positions at scaling
  0115E FA6211          JP      M,1162H         ;If too many positions
                                                ;generated then leave A as it is
  01161 AF              XOR     A               ;else set A = 0
  01162 C5              PUSH    BC              ;Save position counters
  01163 F5              PUSH    AF              ;Save number of positions
                                                ;generated too many
  01164 FC180F          CALL    M,0F18H         ;X = X / 10 , A + 1
  01167 FA6411          JP      M,1164H         ;Reverse scaling until the
                                                ;requested number of positions
                                                ;behind the decimal point is
                                                ;reached
  0116A C1              POP     BC              ;B = neagative number of
                                                ;positions generated too many
  0116B 7B              LD      A,E             ;A = exponent offset
  0116C 90              SUB     B               ;+ number of positions
                                                ;generated too many
  0116D C1              POP     BC              ;Restore position counters
  0116E 5F              LD      E,A             ;E = exponent offset ( < 0!)
  0116F 82              ADD     A,D             ;Is the exponent offset + the
                                                ;maximum number of requested
                                                ;positions before the decimal
                                                ;point < 0 ? (this means no
                                                ;positions before the decimal
                                                ;point)
```

```
  01170 78              LD      A,B             ;A = number of positions
                                                ;before the decimal point
  01171 FA7F11          JP      M,117FH         ;Yes: continue at 117FH

  01174 92              SUB     D               ;A = number of positions before
                                                ;decimal point - maximum number
                                                ;of positions before decimal
                                                ;point
  01175 93              SUB     E               ;- exponent offset
  01176 F46912          CALL    P,1269H         ;Set corresponding number
                                                ;of leading zeroes in buffer
  01179 C5              PUSH    BC              ;Saev position counters
  0117A CD7D12          CALL    127DH           ;Compute decimal point
                                                ;position and counter for
                                                ;thousands separation
  0117D 1811            JR      1190H           ;continue at 1190H

; No positions before decimal point present (only a fractional part, see 116EH)

  0117F CD6912          CALL    1269H           ;Simulate requested field
                                                ;length before decimal point
                                                ;by using leading zeroes
  01182 79              LD      A,C             ;A = number of requested
                                                ;positions behind the decimal
                                                ;point + 1
  01183 CD9412          CALL    1294H           ;Set decinal point, C = B
  01186 4F              LD      C,A             ;Write field length after
                                                ;decimal point back to C
  01187 AF              XOR     A               ;A = 0
  01188 92              SUB     D               ;A = A - maximum number of
                                                ;positions
  01189 93              SUB     E               ;+ exponent offset (+ because
                                                ;E < 0)
  0118A CD6912          CALL    1269H           ;Set corresponding number of
                                                ;trailing zeroes after the
                                                ;decimal point
  0118D C5              PUSH    BC              ;Save position counters
  0118E 47              LD      B,A             ;B = 0 (A = 0 because of 1269H)
  0118F 4F              LD      C,A             ;C = 0: use no decimal point
                                                ;and no separation of thousands
  01190 CDA412          CALL    12A4H           ;Generate unformatted string
  01193 C1              POP     BC              ;Restore position counters
  01194 B1              OR      C               ;A = requested number of
                                                ;positions behing the decimal
                                                ;point.
                                                ;Any positions behind the
                                                ;decimal point specified ?
  01195 2003            JR      NZ,119AH        ;Yes: leave buffer pointer as
                                                ;it is (buffer pointer points
                                                ;at the last string char! )
  01197 2AF340          LD      HL,(40F3H)      ;No: set buffer pointer to the
                                                ;position of the decimal
                                                ;point (= end of string!)
  0119A 83              ADD     A,E             ;A = number of requested
                                                ;positions behind the decimal
                                                ;point + 1 + exponent offset
  0119B 3D              DEC     A               ;Adjust because of + 1
  0119C F46912          CALL    P,1269H         ;Set corresponding number of
                                                ;trailing zeroes after the
                                                ;decimal point
  0119F 50              LD      D,B             ;D = number of requested
                                                ;positions before decimal point
  011A0 C3BF10          JP      10BFH           ;Continue at 10BFH


; INT number using exponential format
; For this, X has to be converted to SNG format

  011A3 E5              PUSH    HL              ;Save buffer pointer
  011A4 D5              PUSH    DE              ;Save format byte
  011A5 CDCC0A          CALL    0ACCH           ;X = CSNG ( X )
```

```
  011A8 D1              POP     DE              ;Restore format byte
  011A9 AF              XOR     A               ;A = 0 (Z-flag = 1 because
                                                ;X in SNG format)
; SNG or DBL number using exponential format
                                                ;SNG number ?
  011AA CAB011          JP      Z,11B0H         ;Yes: continue at 11B0H

  011AD 1E10            LD      E,10H           ;E = maximum number of
                                                ;positions (16)
  011AF 011E06          LD      BC,061EH        ;--
* 011B0   1E06          LD      E,06H           ;E = maximum number of
                                                ;positions (16)
  011B2 CD5509          CALL    0955H           ;TEST2. X = 0 ?
  011B5 37              SCF                     ;C-flag = 1
  011B6 C40112          CALL    NZ,1201H        ;No, scale X to 6 (SNG) or
                                                ;16 (DBL) positions
  011B9 E1              POP     HL              ;Restore buffer pointer
  011BA C1              POP     BC              ;Restore position counters
  011BB F5              PUSH    AF              ;Save exponent offset
  011BC 79              LD      A,C             ;A = number of positions behind
                                                ;the decimal point
  011BD B7              OR      A               ;Any positions behind decimal
                                                ;point specified ?
  011BE F5              PUSH    AF              ;Save number of positions
  011BF C4160F          CALL    NZ,0F16H        ;Yes: A - 1 (because of decimal
                                                ;point)
  011C2 80              ADD     A,B             ;+ number of positions in front
                                                ;of the decimal point
  011C3 4F              LD      C,A             ;C = total length
  011C4 7A              LD      A,D             ;A = format byte
  011C5 E604            AND     04H             ;Print sign behind number ?
  011C7 FE01            CP      01H             ;No: C-flag = 1
  011C9 9F              SBC     A,A             ;If not then A = FFH (-1)
  011CA 57              LD      D,A             ;D = A
  011CB 81              ADD     A,C             ;A = total length, left alone
                                                ;if sign is to printed in front
                                                ;else decrement it by 1
  011CC 4F              LD      C,A             ;C = total length
  011CD 93              SUB     E               ;A = requested total length
                                                ;of string - maximum number
                                                ;of positions
  011CE F5              PUSH    AF              ;Save difference
  011CF C5              PUSH    BC              ;Save total length
                                                ;Is the requested number of
                                                ;positions < generated number
                                                ;of positions
  011D0 FC180F          CALL    M,0F18H         ;Yes: X = X / 10, A - 1
  011D3 FAD011          JP      M,11D0H         ;Scale X according to
                                                ;difference
  011D6 C1              POP     BC              ;Restore total length
  011D7 F1              POP     AF              ;Restore difference
  011D8 C5              PUSH    BC              ;Save total length
  011D9 F5              PUSH    AF              ;Save difference
                                                ;Total length > number of
                                                ;positions ?
  011DA FADE11          JP      M,11DEH         ;Yes: continue at 11DEH

  011DD AF              XOR     A               ;A = 0

  011DE 2F              CPL                     ;A = positive difference
  011DF 3C              INC     A
  011E0 80              ADD     A,B             ;+ number of requested
                                                ;positions on front of
                                                ;decimal point
  011E1 3C              INC     A               ;+ 1
  011E2 82              ADD     A,D             ;-1 if case of sign being
                                                ;printed in front of the number
  011E3 47              LD      B,A             ;= decomal point position
  011E4 0E00            LD      C,00H           ;Set no thousands separation
  011E6 CDA412          CALL    12A4H           ;Generate unformatted string
  011E9 F1              POP     AF              ;Restore position difference
```

```
  011EA F47112           CALL    P,1271H          ;Use corresponding number
                                                  ;of trailing zeroes
  011ED C1               POP     BC               ;Restore position counters
  011EE F1               POP     AF               ;A = requested number of
                                                  ;positions behind the decimal
                                                  ;point.
                                                  ;Any positions requested ?
  011EF CC2F09           CALL    Z,092FH          ;No: buffer pointer - 1
                                                  ;(remove decimal point again)
  011F2 F1               POP     AF               ;Restore exponent offset
                                                  ;X = 0 ? (see 11B5H)
  011F3 3803             JR      C,11F8H          ;Yes: 10-exponent is also 0
                                                  ;No:
  011F5 83               ADD     A,E              ;Add maximum number of
                                                  ;positions
  011F6 90               SUB     B                ;subtract number of already
                                                  ;generated positions in front
                                                  ;of decimal point
  011F7 92               SUB     D                ;Reverse addition of 11E2H
                                                  ;= 10-exponent
  011F8 C5               PUSH    BC               ;Save number of positions in
                                                  ;front of decimal point
  011F9 CD7410           CALL    1074H            ;Use 10-exponent
  011FC EB               EX      DE,HL            ;HL = end pointer
  011FD D1               POP     DE               ;D = number of requested
                                                  ;positions
  011FE C3BF10           JP      10BFH            ;Continue at 10BFH


; Scaling:
; Scale X to 6 (SNG) or 16 (DBL) positions in front of decimal point
; O: A = 10-exponent offset

  01201 D5               PUSH    DE               ;Save DE
  01202 AF               XOR     A                ;Exponent offset = 00H
  01203 F5               PUSH    AF               ;Save exponent offset
  01204 E7               RST     20H              ;TSTTYP
                                                  ;X in SNG format ?
  01205 E22212           JP      PO,1222H         ;Yes: continue at 1222H

; X in DBL format

  01208 3A2441           LD      A,(4124H)        ;A = Exp (X)
  0120B FE91             CP      91H              ;X >= 2 ^ 16 ?
  0120D D22212           JP      NC,1222H         ;Yes: continue at 1222H

  01210 116413           LD      DE,1364H         ;DE -> 1D+10
  01213 212741           LD      HL,4127H         ;HL -> Y
  01216 CDD309           CALL    09D3H            ;(HL) = (DE): Y = 1D+10
  01219 CDA10D           CALL    0DA1H            ;X = X * Y - X * 1D+10
  0121C F1               POP     AF               ;Restore exponent offset
  0121D D60A             SUB     0AH              ;Exponent offset - 10
                                                  ;(shifted 10 decimal positions)
  0121F F5               PUSH    AF               ;Save exponent offset
  01220 18E6             JR      1208H            ;Repeat until X >= 2 ^ 16

; X in SNG format / X >= 65536 when in DBL format

  01222 CD4F12           CALL    124FH            ;Divide X by 10 until
                                                  ;X < 1E+6 (SNG) or
                                                  ;X < 1D+16 (DBL)
; X is now < 1E+6 (SNG) or < 1D+16 (DBL)

  01225 E7               RST     20H              ;TSTTYP
                                                  ;X in DBL format ?
  01226 300B             JR      NC,1233H         ;Yes: continue at1233H

; X has SNG format

  01228 014391           LD      BC,9143H         ;BCDE = 1E+5
  0122B 11F94F           LD      DE,4FF9H
```

```
  0122E CD0C0A          CALL    0A0CH           ;CP X,BCDE = CP X,1E+5
  01231 1806            JR      1239H           ;Continue at 1239H

; X has DBL format

  01233 116C13          LD      DE,136CH        ;DE -> 1D+15
  01236 CD490A          CALL    0A49H           ;CP X,(DE) = CP X,1D+15
                                                ;X >= 1D+15 (DBL) ?
                                                ;X >= 1E+5 (SNG) ?
  01239 F24B12          JP      P,124BH         ;Yes: done

  0123C F1              POP     AF              ;Restore exponent offset
  0123D CD0B0F          CALL    0F0BH           ;X = X + 10 (SNG,DBL), A - 1
  01240 F5              PUSH    AF              ;Save exponent offset
  01241 18E2            JR      1225H           ;Repeat until X >= 1E+5 (SNG)
                                                              X >= 1D+15 (DBL)
; X >= 1E+6 (SNG) or X >= 1D+16 (DBL)
; (Continuation of 124FH)

  01243 F1              POP     AF              ;Restore exponent offset
  01244 CD180F          CALL    0F18H           ;X = X / 10, A + 1
  01247 F5              PUSH    AF              ;Save exponent offset
  01248 CD4F12          CALL    124FH           ;Test again

; Scaling finished
;
;  1E+5 <= X < 1E+6 (SNG)
; 1D+15 <= X < 1D+16 (DBL)

  0124B F1              POP     AF              ;Restore exponent offset
  0124C B7              OR      A               ;C-flag = 0 (for 11B5H)
  0124D D1              POP     DE              ;Restore DE
  0124E C9              RET

; Divide X by 10 until X < 1E+6 (SNG) or X < 1D+16 (DBL)

  0124F E7              RST     20H             ;TSTTYP
                                                ;X in DBL format ?
  01250 EA5E12          JP      PE,125EH        ;Yes: continue at 125EH

; X has SNG format

  01253 017494          LD      BC,9474H        ;BCDE = 1E+6
  01256 11F823          LD      DE,23F8H
  01259 CD0C0A          CALL    0A0CH           ;CP X,BCDE = CP X,1E+6
  0125C 1806            JR      1264H           ;Continue at 1264H

; X has DBL format

  0125E 117413          LD      DE,1374H        ;DE -> 1D+16
  01261 CD490A          CALL    0A49H           ;CP X,(DE) = CP X, 1D+16

  01264 E1              POP     HL              ;HL = RET address
                                                ;X >= 1E+6 (SNG) ?
                                                ;X >= 1D+16 (DBL) ?
  01265 F24312          JP      P,1243H         ;Yes: continue at 1243H

  01268 E9              JP      (HL)            ;RET


; Write A zeroes into buffer from (HL) onwards
; (positions after the decimal point)

  01269 B7              OR      A               ;Counter = 0 ?
  0126A C8              RET     Z               ;Yes: done, return

  0126B 3D              DEC     A               ;Counter - 1
  0126C 3630            LD      (HL),'0'        ;Put a '0' into buffer
  0126E 23              INC     HL              ;Pointer + 1
  0126F 18F9            JR      126AH           ;Done ?
```

```
; Write A zeroes into buffer from (HL) onwards and set ',' and '.'
; (positions before the decimal point)
                                                  ;Counter zero ?
  01271 2004            JR      NZ,1277H         ;No: continue at 1277H

  01273 C8              RET     Z                ;Yes: done, return

  01274 CD9112          CALL    1291H            ;Set ',' and '.'
  01277 3630            LD      (HL),'0'         ;Put a '0' into buffer
  01279 23              INC     HL               ;Pointer + 1
  0127A 3D              DEC     A                ;Counter - 1
  0127B 18F6            JR      1273H            ;Done ?


; Establish decimal point position and counter for thousands separation

  0127D 7B              LD      A,E              ;A = exponent offset
  0127E 82              ADD     A,D              ;+ maimum number of generated
                                                 ;positions
  0127F 3C              INC     A                ;+ 1
  01280 47              LD      B,A              ;= decimal point position
                                                 ;(=number of positions before
                                                 ;the decimal point)
                                                 ;Ccompute counter for thousands
                                                 ;separation
  01281 3C              INC     A                ;+ 1
  01282 D603            SUB     03H              ;A = A DIV 3
  01284 30FC            JR      NC,1282H         ;(Integer division)

  01286 C605            ADD     A,05H            ;+5
  01288 4F              LD      C,A              ;Results in counter for
                                                 ;thousands separation
  01289 3AD840          LD      A,(40D8H)        ;A = format byte
  0128C E640            AND     40H              ;Thousands separation required?
  0128E C0              RET     NZ               ;Yes: B and C are ok, return

  0128F 4F              LD      C,A              ;Set C to 0
  01290 C9              RET


; Set ',' and '.'
;
; I: B = number of remaining positions before the decimal point (until
;        decimal point)
;    C = number of remaining digits until next thousand separation
;    C = 0: do not separate thousands

  01291 05              DEC     B                ;Positions before decimal
                                                 ;point - 1
                                                 ;Decimal point reached ?
  01292 2008            JR      NZ,129CH         ;No: set ','

  01294 362E            LD      (HL),'.'         ;Put '.' into buffer

; Save buffer pointer to decimal point
; Do not separate thousands

  01296 22F340          LD      (40F3H),HL       ;Save buffer pointer
  01299 23              INC     HL               ;Pointer + 1
  0129A 48              LD      C,B              ;C = 0 (no thousand separation)
  0129B C9              RET


; Decimal point not reached yet: insert thousands separation

  0129C 0D              DEC     C                ;Next thousand position
                                                 ;reached ?
  0129D C0              RET     NZ               ;No: done, return

  0129E 362C            LD      (HL),','         ;Yes, put ',' into buffer
```

```
  012A0 23                INC     HL              ;Pointer + 1
  012A1 0E03              LD      C,03H           ;Counter = 3 for next
                                                  ;thousands position
  012A3 C9                RET
```

```
; Convert floating point number to unformatted string

  012A4 D5                PUSH    DE              ;Save DE
  012A5 E7                RST     20H             ;TSTTYP
                                                  ;SNG ?
  012A6 E2EA12            JP      PO,12EAH        ;Yes: continue at 12EAH
```

```
; X in DBL format (and 1D+15 <= X < 1D+16 !)
; Generate unformatted string with 17 positions
; (10 positions using DBL-mantissas, 2 positions using SNG-mantissas
; and 5 positions with INT-format)

  012A9 C5                PUSH    BC              ;Save position counters
  012AA E5                PUSH    HL              ;Save buffer pointer
  012AB CDFC09            CALL    09FCH           ;Y = X
  012AE 217C13            LD      HL,137CH        ;HL -> Constant 0.5 (DBL)
  012B1 CDF709            CALL    09F7H           ;X = (HL) = 0.5
  012B4 CD770C            CALL    0C77H           ;X = X + Y = 0.5 + Y
                                                  ;(Round up X)
  012B7 AF                XOR     A               ;C-flag = 0
  012B8 CD7B0B            CALL    0B7BH           ;Clear positions behind the
                                                  ;decimal point
  012BB E1                POP     HL              ;Restore buffer pointer
  012BC C1                POP     BC              ;Restore position counters
  012BD 118C13            LD      DE,138CH        ;DE -> DBL-mantissas
  012C0 3E0A              LD      A,0AH           ;A = counter (10 DBL-mantissas
                                                  ;from (DE) onwards)
  012C2 CD9112            CALL    1291H           ;Set ',' and '.'
  012C5 C5                PUSH    BC              ;Save position counters
  012C6 F5                PUSH    AF              ;Save mantissa counter
  012C7 E5                PUSH    HL              ;Save buffer pointer
  012C8 D5                PUSH    DE              ;Save mantissa pointer
  012C9 062F              LD      B,2FH           ;B = ASCII '0' - 1
  012CB 04                INC     B               ;Next digit
  012CC E1                POP     HL              ;HL = mantissa pointer
  012CD E5                PUSH    HL              ;Save it again
  012CE CD480D            CALL    0D48H           ;X = X - (HL): subtract
                                                  ;mantissa.
                                                  ;Underflow ?
  012D1 30F8              JR      NC,12CBH        ;No: next digit

  012D3 E1                POP     HL              ;Yes: restore mantissa pointer
  012D4 CD360D            CALL    0D36H           ;X = X + (HL): reverse last
                                                  ;subtract
  012D7 EB                EX      DE,HL           ;DE = mantissa pointer
  012D8 E1                POP     HL              ;HL = buffer pointer
  012D9 70                LD      (HL),B          ;Insert digit in buffer
  012DA 23                INC     HL              ;Update buffer pointer to next
                                                  ;position
  012DB F1                POP     AF              ;Restore maintissa counter
  012DC C1                POP     BC              ;Restore position counters
  012DD 3D                DEC     A               ;Mantissa counter - 1
  012DE 20E2              JR      NZ,12C2H        ;Next decimal position

  012E0 C5                PUSH    BC              ;Save position counters
  012E1 E5                PUSH    HL              ;Save buffer pointer
  012E2 211D41            LD      HL,411DH        ;HL -> X (DBL)
  012E5 CDB109            CALL    09B1H           ;X = BCDE = (HL) (SNG)
                                                  ;Shift the remaining LSBs of
                                                  ;the DBL number into X as a
                                                  ;SNG number
  012E8 180C              JR      12F6H           ;The remaining decimal
                                                  ;positions are processed in
                                                  ;SNG format (because now
```

```
                                                   ;X < 1D+6)

; X in SNG format (and 1E+5 <= X < 1E+6 !)
; Generate unformatted string with 7 positions
; (2 positions using SNG-mantissas and the remaining 5 positions with
; INT-format)

  012EA C5              PUSH    BC              ;Save position counters
  012EB E5              PUSH    HL              ;Save buffer pointers
  012EC CD0807          CALL    0708H           ;X = X + 0.5 (adjust mantissa)
  012EF 3C              INC     A               ;A <> 0 (for 0AFBH)
  012F0 CDFB0A          CALL    0AFBH           ;Clear all positions behind
                                                ;the decimal point
  012F3 CDB409          CALL    09B4H           ;X = BCDE
                                                ;(BCDE is the result of 0AFBH)
  012F6 E1              POP     HL              ;Restore buffer pointer
  012F7 C1              POP     BC              ;Restore position counters
  012F8 AF              XOR     A               ;C-flag = 0
  012F9 11D213          LD      DE,13D2H        ;DE -> mantissas
  012FC 3F              CCF                     ;C-flag = 1 at first itteration
                                                ;after that C-flag = 0
  012FD CD9112          CALL    1291H           ;Set ',' and '.'
  01300 C5              PUSH    BC              ;Save position counters
  01301 F5              PUSH    AF              ;Save repeat flag
  01302 E5              PUSH    HL              ;Save buffer pointer
  01303 D5              PUSH    DE              ;Save mantissa pointer
  01304 CDBF09          CALL    09BFH           ;BCDE = X
  01307 E1              POP     HL              ;Mantissa pointer back to HL
  01308 062F            LD      B,2FH           ;B = '0' - 1
  0130A 04              INC     B               ;next digit
  0130B 7B              LD      A,E             ;CDE = CDE - (HL):
  0130C 96              SUB     (HL)            ;Subtract mantissa
  0130D 5F              LD      E,A
  0130E 23              INC     HL
  0130F 7A              LD      A,D
  01310 9E              SBC     A,(HL)
  01311 57              LD      D,A
  01312 23              INC     HL
  01313 79              LD      A,C
  01314 9E              SBC     A,(HL)
  01315 4F              LD      C,A
  01316 2B              DEC     HL              ;HL -> start of mantissa
  01317 2B              DEC     HL              ;Underflow ?
  01318 30F0            JR      NC,130AH        ;No: next digit

  0131A CDB707          CALL    07B7H           ;CDE= CDE + (HL):
                                                ;reverse last subtract
  0131D 23              INC     HL              ;Mantissa pointer + 1
  0131E CDB409          CALL    09B4H           ;X = BCDE: new value to X
  01321 EB              EX      DE,HL           ;DE = mantissa pointer
  01322 E1              POP     HL              ;Restore buffer pointer
  01323 70              LD      (HL),B          ;Store digit
  01324 23              INC     HL              ;Buffer pointer + 1
  01325 F1              POP     AF              ;Restore repeat flag
  01326 C1              POP     BC              ;Restore position counters
                                                ;Repeat ?
  01327 38D3            JR      C,12FCH         ;Yes: back to 12FCH (there are
                                                ;only 2 mantissas in the SNG
                                                ;format)
  01329 13              INC     DE              ;Mantissa pointer + 2
  0132A 13              INC     DE              ;(for INT processing)
  0132B 3E04            LD      A,04H           ;Process 4 mantissas
                                                ;(the first mantissa with 10000
                                                ;is also in the SNG format)
  0132D 1806            JR      1335H           ;Process remaining 4 digits
                                                ;in INT-format
; X in INT-format (and 0 <= X < 32768 !)
; Generate unfomratted string with 5 positions

  0132F D5              PUSH    DE              ;Save DE
  01330 11D813          LD      DE,13D8H        ;DE -> Mantissas
```

```
  01333 3E05           LD     A,05H          ;5 Mantissas
  01335 CD9112         CALL   1291H          ;Set '.' and ','
  01338 C5             PUSH   BC             ;Save position counters
  01339 F5             PUSH   AF             ;Save mantissa counter
  0133A E5             PUSH   HL             ;Save buffer pointer
  0133B EB             EX     DE,HL          ;HL = mantissa pointer
  0133C 4E             LD     C,(HL)         ;BC = mantissa
  0133D 23             INC    HL
  0133E 46             LD     B,(HL)
  0133F C5             PUSH   BC             ;Save mantissa
  01340 23             INC    HL             ;HL -> next mantissa
  01341 E3             EX     (SP),HL        ;Save mantissa pointer,
                                             ;HL = mantissa
  01342 EB             EX     DE,HL          ;DE = mantissa
  01343 2A2141         LD     HL,(4121H)     ;HL = X
  01346 062F           LD     B,2FH          ;B = ASCII '0' - 1

  01348 04             INC    B              ;Next digit
  01349 7D             LD     A,L            ;HL = HL - DE
  0134A 93             SUB    E
  0134B 6F             LD     L,A
  0134C 7C             LD     A,H
  0134D 9A             SBC    A,D
  0134E 67             LD     H,A            ;Underflow ?
  0134F 30F7           JR     NC,1348H       ;No: next digit
                                             ;Yes:
  01351 19             ADD    HL,DE          ;Reverse subtract
  01352 222141         LD     (4121H),HL     ;and store new value in X
  01355 D1             POP    DE             ;Restore mantissa pointer
  01356 E1             POP    HL             ;Restore buffer pointer
  01357 70             LD     (HL),B         ;Insert digit
  01358 23             INC    HL             ;Buffer pointer + 1
  01359 F1             POP    AF             ;Restore mantissa counter
  0135A C1             POP    BC             ;Restore position counters
  0135B 3D             DEC    A              ;Mantissa counter - 1
  0135C 20D7           JR     NZ,1335H       ;Next mantissa

  0135E CD9112         CALL   1291H          ;Set ',' and '.'
  01361 77             LD     (HL),A         ;Terminate string with 00H
  01362 D1             POP    DE             ;Restore DE
  01363 C9             RET


; Floating point constants

  01364 00             DEFB   00H            ;1D+10
  01365 00             DEFB   00H
  01366 00             DEFB   00H
  01367 00             DEFB   00H
  01368 F9             DEFB   F9H            ;1E+10
  01369 02             DEFB   02H
  0136A 15             DEFB   15H
  0136B A2             DEFB   A2H

  0136C FD             DEFB   FDH            ;1D+15
  0136D FF             DEFB   FFH
  0136E 9F             DEFB   9FH
  0136F 31             DEFB   31H
  01370 A9             DEFB   A9H            ;1E+15
  01371 5F             DEFB   5FH
  01372 63             DEFB   63H
  01373 B2             DEFB   B2H

  01374 FE             DEFB   FEH            ;1D+16
  01375 FF             DEFB   FFH
  01376 03             DEFB   03H
  01377 BF             DEFB   BFH
  01378 C9             DEFB   C9H            ;1E+16
  01379 1B             DEFB   1BH
  0137A 0E             DEFB   0EH
  0137B B6             DEFB   B6H
```

```
  0137C 00               DEFB    00H             ;0.5 (DBL)
  0137D 00               DEFB    00H
  0137E 00               DEFB    00H
  0137F 00               DEFB    00H
  01380 00               DEFB    00H             ;0.5 (SNG)
  01381 00               DEFB    00H
  01382 00               DEFB    00H
  01383 80               DEFB    80H

  01384 00               DEFB    00H             ;1D+16
  01385 00               DEFB    00H
  01386 04               DEFB    04H
  01387 BF               DEFB    BFH
  01388 C9               DEFB    C9H             ;1E+16
  01389 1B               DEFB    1BH
  0138A 0E               DEFB    0EH
  0138B B6               DEFB    B6H

; Fixed point constants (mantissas for number conversion)

;       LSB         MSB

                                 ;1000000000000000

  0138C 0080C6A47E8D03   DEFB    00H,80H,C6H,A4H,7EH,8DH,03H     ;1D+15

                                 ;100000000000000

  01393 00407A10F35A00   DEFB    00H,40H,7AH,10H,F3H,5AH,00H     ;1D+14

                                 ;10000000000000

  0139A 00A0724E180900   DEFB    00H,A0H,72H,4EH,18H,09H,00H     ;1D+13

                                 ;1000000000000

  013A1 0010A5D4E80000   DEFB    00H,10H,A5H,D4H,E8H,00H,00H     ;1D+12

                                 ;100000000000

  013A8 00E87648170000   DEFB    00H,E8H,76H,48H,17H,00H,00H     ;1D+11

                                 ;10000000000

  013AF 00E40B54020000   DEFB    00H,E4H,0BH,54H,02H,00H,00H     ;1D+10

                                 ;1000000000

  013B6 00CA9A3B000000   DEFB    00H,CAH,9AH,3BH,00H,00H,00H     ;1D+9

                                 ;100000000

  013BD 00E1F505000000   DEFB    00H,E1H,F5H,05H,00H,00H,00H     ;1D+8

                                 ;10000000

  013C4 80969800000000   DEFB    80H,96H,98H,00H,00H,00H,00H     ;1D+7

                                 ;1000000

  013CB 40420F00000000   DEFB    40H,42H,0FH,00H,00H,00H,00H     ;1D+6

                                 ;100000

  013D2 A08601           DEFB    A0H,86H,01H                     ;1E+5

                                 ;10000

  013D5 102700           DEFB    10H,27H,00H                     ;1E+4
```

```
                                ;1000

  013D8 E803            DEFB    E8H,03H                         ;INT

                                ;100

  013DC 6400            DEFB    64H,00H

                                ;10

  013DE 0A00            DEFB    0AH,00H

                                ;1

  013E0 0100            DEFB    01H,00H


; SUB for SQR and ATN
; Negate the result of the routine (X = -X)

  013E2 218209          LD      HL,0982H        ;HL -> Routine for X = -X
  013E5 E3              EX      (SP),HL         ;Save HL as RET address on
                                                ;stack
  013E6 E9              JP      (HL)            ;Back to caller (SQR or ATN)


; X = SQR ( X ) = X ^ 0.5
; -----------------------
;

  013E7 CDA409          CALL    09A4H           ;(SP) = X
  013EA 218013          LD      HL,1380H        ;HL -> constant 0.5
  013ED CDB109          CALL    09B1H           ;X = BCDE = (HL)
  013F0 1803            JR      13F5H           ;Compute (SP) ^ X

; X = (SP) ^ X

  013F2 CDB10A          CALL    0AB1H           ;X = CSNG (X) (Exponent)
  013F5 C1              POP     BC              ;BCDE = (SP) (Base)
  013F6 D1              POP     DE
  013F7 CD5509          CALL    0955H           ;TEST2
  013FA 78              LD      A,B             ;A = Exp (Base)
                                                ;Exponent zero ?
  013FB 283C            JR      Z,1439H         ;Yes: compute Exp(0) = 1
                                                ;Exponent > 0
  013FD F20414          JP      P,1404H         ;Yes: continue at 1404H

  01400 B7              OR      A               ;Base = zero ?
                                                ;(with negative exponent)
  01401 CA9A19          JP      Z,199AH         ;Yes: ?/0 Error

  01404 B7              OR      A               ;Base = zero ?
                                                ;(with negative exponent)
  01405 CA7907          JP      Z,0779H         ;Yes: ?/0 Error

  01408 D5              PUSH    DE              ;Save Base
  01409 C5              PUSH    BC
  0140A 79              LD      A,C             ;A = MSB (Base)
  0140B F67F            OR      7FH             ;Test sign
  0140D CDBF09          CALL    09BFH           ;BCDE = X
                                                ;Positive Base ?
  01410 F22114          JP      P,1421H         ;Yes: continue at 1421H

  01413 D5              PUSH    DE              ;Save Exponent
  01414 C5              PUSH    BC
  01415 CD400B          CALL    0B40H           ;X = INT (X) = INT ( Exponent)
  01418 C1              POP     BC              ;Restore exponent
  01419 D1              POP     DE
  0141A F5              PUSH    AF              ;Save LSB (X)
  0141B CD0C0A          CALL    0A0CH           ;CP X,BCDE
                                                ;CP INT(Exponent), Exponent
```

```
                                                    ;Exponent an integer ?
  0141E E1               POP     HL                ;Restore LSB (X)
  0141F 7C               LD      A,H               ;A = LSB (X) = INT (Exponent)
                                                    ;(because exponent < 88)
  01420 1F               RRA                       ;C-flag = A, bit 0 (lowest bit
                                                    ;of the exponent)
                                                    ;Exponent odd ?
  01421 E1               POP     HL                ;X = (SP) = Base
  01422 222341           LD      (4123H),HL
  01425 E1               POP     HL
  01426 222141           LD      (4121H),HL
                                                    ;Exponent odd ?
  01429 DCE213           CALL    C,13E2H           ;Yes: negate result when
                                                    ;exponent is odd with negative
                                                    ;base
                                                    ;Exponent an integer ?
  0142C CC8209           CALL    Z,0982H           ;Yes: make base positive when
                                                    ;exponent is integer.
  0142F D5               PUSH    DE                ;Save Exponent
  01430 C5               PUSH    BC
  01431 CD0908           CALL    0809H             ;X = LOG (X) (logaritm of Base)
  01434 C1               POP     BC                ;Restore Exponent
  01435 D1               POP     DE
  01436 CD4708           CALL    0847H             ;X = BCDE * X
                                                    ;  = Exponent * LOG (Base)
                                                    ;Now compute the following:
                                                    ;EXP ( Exponent * LOG (Base) )
; X = EXP ( X )
; -------------
; Only computable for -88.7228 <= X <= 87.3365

  01439 CDA409           CALL    09A4H             ;(SP) = X: Save argument
  0143C 013881           LD      BC,8138H          ;BCDE = 1/4427 = 1 / LOG(2)
  0143F 113BAA           LD      DE,0AA3BH
  01442 CD4708           CALL    0847H             ;X = BCDE * argument
                                                    ;  = argument / LOG(2)
  01445 3A2441           LD      A,(4124H)         ;A = Exp (X)
  01448 FE88             CP      88H               ;Exp (X) >= 88H  ?
                                                    ;means: X >= 2 ^ 8  ?
                                                    ;means: argument/LOG(2) > 127
                                                    ;(or argument/LOG(2) < -128)  ?
                                                    ;means: argument > 88.0297
                                                    ;(or argument < -88.7228) ?
  0144A D23109           JP      NC,0931H          ;Yes: ?OV Error if
                                                    ;argument > 88.0297,
                                                    ;Result = 0 if
                                                    ;argument < 88.7228)
  0144D CD400B           CALL    0B40H             ;A = X = INT (X)
                                                    ;( X = INT( argument/LOG(2) ) )
                                                    ;(   = 2-exponent of result )
  01450 C680             ADD     A,80H             ;A > 7DH ? (A + 82H > FFH)
  01452 C602             ADD     A,02H             ;means: INT(arg./LOG(2)) > 125?
                                                    ;means X > 87.3365 ?
  01454 DA3109           JP      C,0931H           ;Yes: ?OV Error
                                                    ;No:
  01457 F5               PUSH    AF                ;Save 2-exponent + offset (80H)
                                                    ;+ 2
  01458 21F807           LD      HL,07F8H          ;HL -> Constant 1 (SNG)
  0145B CD0B07           CALL    070BH             ;X = X + HL = X + 1
  0145E CD4108           CALL    0841H             ;X = X * LOG(2)
  01461 F1               POP     AF                ;Restore 2-exponent
  01462 C1               POP     BC                ;BCDE = (SP) = argument
  01463 D1               POP     DE
  01464 F5               PUSH    AF                ;Save 2-exponent
  01465 CD1307           CALL    0713H             ;X = BCDE - X
  01468 CD8209           CALL    0982H             ;X = -X
  0146B 217914           LD      HL,1479H          ;HL -> coefficients
  0146E CDA914           CALL    14A9H             ;Series calculation 2
  01471 110000           LD      DE,0000H          ;DE = 0000H
  01474 C1               POP     BC                ;B = 2-exponent + 2
  01475 4A               LD      C,D               ;C = 00H
```

```
  01476 C34708           JP      0847H           ;X = BCDE * X
                                                 ;  = (2 ^ 2 + 2-exponent) * X

; Coefficients for EXP
; '!' means faculty of the number (3! = 1 * 2 * 3)

  01479 08               DEFB    08H             ;8 coefficients

  0147A 40               DEFB    40H             ;-1.41361E-4 = -1/7076
  0147B 2E               DEFB    2EH             ;approx. -1/5040 = -1/7!
  0147C 94               DEFB    94H
  0147D 74               DEFB    74H

  0147E 70               DEFB    70H             ;1.32988E-3 = 1/752
  0147F 4F               DEFB    4FH             ;approx. 1/720 = 1/6!
  01480 2E               DEFB    2EH
  01481 77               DEFB    77H

  01482 6E               DEFB    6EH             ;=8.30136E-3 = -1/120
  01483 02               DEFB    02H                          = -1/5!
  01484 88               DEFB    88H
  01485 7A               DEFB    7AH

  01486 E6               DEFB    E6H             ;0.0416574 = 1/24
  01486 A0               DEFB    A0H                        = 1/4!
  01488 2A               DEFB    2AH
  01489 7C               DEFB    7CH

  0148A 50               DEFB    50H             ;-0.166665 = -1/6
  0148B AA               DEFB    AAH                        = -1/3!
  0148C AA               DEFB    AAH
  0148D 7E               DEFB    7EH

  0148E FF               DEFB    FFH             ;0.5 = 1/2
  0148F FF               DEFB    FFH                  = 1/2!
  01490 7F               DEFB    7FH
  01491 7F               DEFB    7FH

  01492 00               DEFB    00H             ;-1 = -1/1!
  01493 00               DEFB    00H
  01494 80               DEFB    80H
  01495 81               DEFB    81H

  01496 00               DEFB    00H             ;1
  01497 00               DEFB    00H
  01498 00               DEFB    00H
  01499 81               DEFB    81H

; Series calculation 1
; Calculate Taylor-series of the following form:
;
; y = k1*x + k2*x*x*x + k3*x*x*x*x*x... (k1, k2, k3 are coefficients)
;
; I: HL -> coefficients table
;          The first byte of the table indicates the number of coefficients
;          in the table. Then the coefficients follow, in an inverted
;          order (k1 last)
;    X = factor in series (x in the example)
; O: X = result of series calculation (y in the example)

  0149A CDA409           CALL    09A4H           ;(SP) = X
  0149D 11320C           LD      DE,0C32H        ;DE = address of X = X * (SP)
  014A0 D5               PUSH    DE              ;Save as new RET address
  014A1 E5               PUSH    HL              ;Save table pointer
  014A2 CDBF09           CALL    09BFH           ;BCDE = X
  014A5 CD4708           CALL    0847H           ;X = BCDE * X = X * X
  014A8 E1               POP     HL              ;Restore table pointer and
                                                 ;Series calulation 2 using
                                                 ;X * X and multiply result
                                                 ;again with X
; Series calculation 2
```

```
; Calculate Taylor-series of the following form:
;
; y = k1 + k2*x + k3*x*x*x... (k1, k2, k3 are coefficients)
;
; I: HL -> coefficients table
;          The first byte of the table indicates the number of coefficients
;          in the table. Then the coefficients follow, in an inverted
;          order (k1 last)
;    X = factor in series (x in the example)
; O: X = result of series calculation (y in the example)

  014A9 CDA409         CALL    09A4H          ;(SP) = X
  014AC 7E             LD      A,(HL)         ;A = number of coefficients
  014AD 23             INC     HL             ;HL -> 1st number (last
                                              ;coefficient in the series)
  014AE CDB109         CALL    09B1H          ;X = BCDE = (HL) = 1st coeff.
  014B1 06F1           LD      B,0F1H         ;--
* 014B1   F1           POP     AF             ;Restore counter
  014B3 C1             POP     BC             ;BCDE = X
  014B4 D1             POP     DE
  014B5 3D             DEC     A              ;Any more coefficients ?
  014B6 C8             RET     Z              ;No: done, return

  014B7 D5             PUSH    DE             ;Save BCDE
  014B8 C5             PUSH    BC
  014B9 F5             PUSH    AF             ;Save counter
  014BA E5             PUSH    HL             ;Save pointer
  014BB CD4708         CALL    0847H          ;Result = result + BCDE
  014BE E1             POP     HL             ;Restore pointer
  014BF CDC209         CALL    09C2H          ;BCDE = (HL) = coefficient
  014C2 E5             PUSH    HL             ;Save pointer
  014C3 CD1607         CALL    0716H          ;Result = result + BCDE
  014C6 E1             POP     HL             ;Restore pointer
  014C7 18E9           JR      14B2H          ;Calculate next term


; X = RND ( X )
; -------------
; For X > 1: RND ( X ) = INT( RND(0) * INT(X) + 1 )

  014C9 CD7F0A         CALL    0A7FH          ;X = HL = CINT( argument )
  014CC 7C             LD      A,H            ;Argument negative ?
  014CD B7             OR      A
  014CE FA4A1E         JP      M,1E4AH        ;Yes: ?FC Error

  014D1 B5             OR      L              ;Argument = 0 ?
  014D2 CAF014         JP      Z,14F0H        ;Yes: compute RND (0)

  014D5 E5             PUSH    HL             ;Save argument
  014D6 CDF014         CALL    14F0H          ;Compute RND (0)
  014D9 CDBF09         CALL    09BFH          ;BCDE = X = RND (0)
  014DC EB             EX      DE,HL          ;(SP) = BCDE, HL = argument
  014DD E3             EX      (SP),HL
  014DE C5             PUSH    BC
  014DF CDCF0A         CALL    0ACFH          ;X = CSNG (HL)
  014E2 C1             POP     BC             ;BCDE = (SP) = RND (0)
  014E3 D1             POP     DE
  014E4 CD4708         CALL    0847H          ;X = X * BCDE
                                              ;  = argument * RND (0)
  014E7 21F807         LD      HL,07F8H       ;HL -> Constant 1 (SNG)
  014EA CD0B07         CALL    070BH          ;X = X + (HL) = X + 1
  014ED C3400B         JP      0B40H          ;X = CINT (X)


; X = RND ( 0 )
;   = last random number * 0.253514 + 0.022228 (ignore carry)

  014F0 219040         LD      HL,4090H       ;HL -> multiplicator
  014F3 E5             PUSH    HL             ;Save pointer
  014F4 110000         LD      DE,0000H       ;CDE = 000000H
  014F7 4B             LD      C,E            ;(Result in CDE)
```

```
  014F8 2603            LD      H,03H           ;H = byte counter
                                                ;(3 bytes mantissa)
  014FA 2E08            LD      L,08H           ;L = bit counter
                                                ;(8 bits per byte)
  014FC EB              EX      DE,HL           ;Shift CDE 1 bit to the left
  014FD 29              ADD     HL,HL           ;(for multiplication)
  014FE EB              EX      DE,HL
  014FF 79              LD      A,C
  01500 17              RLA
  01501 4F              LD      C,A
  01502 E3              EX      (SP),HL         ;Save counter, restore pointer
  01503 7E              LD      A,(HL)          ;Shift next bit of
  01504 07              RLCA                    ;multiplicator into C-flag
  01505 77              LD      (HL),A
  01506 E3              EX      (SP),HL         ;Save pointer, restore counter
                                                ;Next bit = 1 ?
  01507 D21615          JP      NC,1516H        ;No: continue at 1516H
                                                ;Yes:
  0150A E5              PUSH    HL              ;Save pointer
  0150B 2AAA40          LD      HL,(40AAH)      ;HL = LSBs of last random
                                                ;number
  0150E 19              ADD     HL,DE           ;Add to result
  0150F EB              EX      DE,HL
  01510 3AAC40          LD      A,(40ACH)       ;A = MSB of last random number
  01513 89              ADC     A,C             ;Add to result
  01514 4F              LD      C,A
  01515 E1              POP     HL              ;Restore counter
  01516 2D              DEC     L               ;Bit counter - 1
                                                ;All bits done ?
  01517 C2FC14          JP      NZ,14FCH        ;No: process next bit
                                                ;Yes:
  0151A E3              EX      (SP),HL         ;Save counter, restore pointer
  0151B 23              INC     HL              ;Pointer + 1 (process next byte
                                                ;of the multiplicator)
  0151C E3              EX      (SP),HL         ;Save pointer, restore counter
  0151D 25              DEC     H               ;Byte counter - 1
                                                ;All 3 bytes processed ?
  0151E C2FA14          JP      NZ,14FAH        ;No: process next byte
                                                ;Yes:
  01521 E1              POP     HL              ;Remove pointer from stack
  01522 2165B0          LD      HL,0B065H       ;and add 05B065H (=0.022228)
  01525 19              ADD     HL,DE           ;to mantissa
  01526 22AA40          LD      (40AAH),HL      ;Store new random number
  01529 CDEF0A          CALL    0AEFH           ;Set VT = SNG
  0152C 3E05            LD      A,05H           ;Adjust MSB
  0152E 89              ADC     A,C
  0152F 32AC40          LD      (40ACH),A       ;And store it
  01532 EB              EX      DE,HL           ;CDE = mantissa of new random
                                                ;number
  01533 0680            LD      B,80H           ;Set exponent = 80H
  01535 212541          LD      HL,4125H        ;HL -> sign flag
  01538 70              LD      (HL),B          ;Sign-flag = 80H
                                                ;(generate a positive number
                                                ;at SFLOAT)
  01539 2B              DEC     HL              ;HL -> Exp (X)
  0153A 70              LD      (HL),B          ;Exp (X) = 80H (result < 1)
  0153B 4F              LD      C,A             ;Use MSB in BCDE
  0153C 0600            LD      B,00H           ;LSB of CDEB = 00H
  0153E C36507          JP      0765H           ;Convert CDEB into SNG format
                                                ;and store in X (SFLOAT)

; X = COS ( X )
; -------------
; X = COS (X) = SIN (X + PI / 2). PI = 3.14159

  01541 218B15          LD      HL,158BH        ;HL -> PI/2
  01544 CD0B07          CALL    070BH           ;X = argument + PI/2


; X = SIN ( X )
; -------------
```

```
  01547 CDA409          CALL    09A4H           ;(SP) = argument
  0154A 014983          LD      BC,8349H        ;BCDE = 6.28319 = 2*PI
  0154D 11DB0F          LD      DE,0FDBH
                                                ;Sace X into the range of
                                                ;-1 to 1
  01550 CDB409          CALL    09B4H           ;X = BCDE = 2*PI
  01553 C1              POP     BC              ;BCDE = argument
  01554 D1              POP     DE
  01555 CDA208          CALL    08A2H           ;X = X / BCDE = argument/(2*PI)
  01558 CDA409          CALL    09A4H           ;(SP) = X = argument/(2*PI)
  0155B CD400B          CALL    0B40H           ;X = INT (X)
                                                ;  = INT ( argument / (2*PI) )
  0155E C1              POP     BC              ;BCDE = argument / (2*PI)
  0155F D1              POP     DE
  01560 CD1307          CALL    0713H           ;X = BCDE - X
                                                ;= arg./(2*PI)-INT(arg./(2*PI))
                                                ;(remove multiples of 2*PI)
                                                ;The argument (X) is now
                                                ;between -1 (corresponds to
                                                ;-2*PI) and 1 (corresponds to
                                                ;+2*PI)
  01563 218F15          LD      HL,158FH        ;HL -> 1/4
  01566 CD1007          CALL    0710H           ;X = (HL) - X = 1/4 - X
  01569 CD5509          CALL    0955H           ;TEST2: X >= 0 ? (arg. > PI/2)
  0156C 37              SCF                     ;C-flag = 1
  0156D F27715          JP      P,1577H         ;Yes: continue at 1577H
                                                ;No:
  01570 CD0807          CALL    0708H           ;X = X + 1/2
  01573 CD5509          CALL    0955H           ;TEST2: X >= 0 ? (arg < PI/2)
  01576 B7              OR      A               ;C-flag = 0 (for 1582H)
  01577 F5              PUSH    AF              ;Save flags
  01578 F48209          CALL    P,0982H         ;Yes: X = -X
  0157B 218F15          LD      HL,158FH        ;HL -> 1/4
  0157E CD0B07          CALL    070BH           ;X = X + (HL) = X + 1/4
  01581 F1              POP     AF              ;Restore flags
  01582 D48209          CALL    NC,0982H        ;if (1/4-X) > 0 then X = -X
  01585 219315          LD      HL,1593H        ;HL -> coefficients
  01588 C39A14          JP      149AH           ;Calculate series 1


  0158B DB              DEFB    DBH             ;Constant 1.5708 = PI / 2
  0158C 0F              DEFB    0FH
  0158D 49              DEFB    49H
  0158E 81              DEFB    81H

  0158F 00              DEFB    00H             ;Constant 0.25 = 1/4
  01590 00              DEFB    00H
  01591 00              DEFB    00H
  01592 7F              DEFB    7FH


; Coefficient table for SIN and COS
; '!' means faculty of the number (3! = 1 * 2 * 3)

  01593 05              DEFB    05H             ;5 coefficients

  01594 BA              DEFB    BAH             ;39.7107
  01595 D7              DEFB    D7H             ;= ((2*PI) ^ 9) / 9!
  01596 1E              DEFB    1EH
  01597 86              DEFB    86H

  01598 64              DEFB    64H             ;-76.575
  01599 26              DEFB    26H             ;= - ((2*PI) ^ 7) / 7!
  0159A 99              DEFB    99H
  0159B 87              DEFB    87H

  0159C 58              DEFB    58H             ;81.6022
  0159D 34              DEFB    34H             ;= ((2*PI) ^ 5) / 5!
  0159E 23              DEFB    23H
  0159F 87              DEFB    87H
```

```
  015A0 E0              DEFB    E0H             ;-41.3417
  015A1 5D              DEFB    5DH             ;= - ((2*PI) ^ 3) / 3!
  015A2 A5              DEFB    A5H
  015A3 86              DEFB    86H

  015A4 DA              DEFB    DAH             ;6.28319 = 2*PI
  015A5 0F              DEFB    0FH             ;= ((2*PI) ^ 1) / 1!
  015A6 49              DEFB    49H
  015A7 83              DEFB    83H


; X = TAN ( X )
; -------------
; X = TAN (X) = SIN (X) / COS (X)

  015A8 CDA409          CALL    09A4H           ;(SP) = argument
  015AB CD4715          CALL    1547H           ;X = SIN (argument)
  015AE C1              POP     BC              ;BCHL = argument
  015AF E1              POP     HL
  015B0 CDA409          CALL    09A4H           ;(SP) = X = SIN (argument)
  015B3 EB              EX      DE,HL           ;BCDE = argument
  015B4 CDB409          CALL    09B4H           ;X = BCDE = argument
  015B7 CD4115          CALL    1541H           ;X = COS (argument)
  015BA C3A008          JP      08A0H           ;Result = (SP) / X
                                                ;= SIN (arg.) / COS (arg.)

; X = ATN ( X )
; -------------

  015BD CD5509          CALL    0955H           ;TEST2: argument < 0 ?
  015C0 FCE213          CALL    M,13E2H         ;Yes: negate restult afterwards
  015C3 FC8209          CALL    M,0982H         ;and continue processing with
                                                ;positive argument
  015C6 3A2441          LD      A,(4124H)       ;A = Exp (argument)
  015C9 FE81            CP      81H             ;Argument < 1 ?
  015CB 380C            JR      C,15D9H         ;Yes: continue at 15D9H

  015CD 010081          LD      BC,8100H        ;BCDE = 1.0
  015D0 51              LD      D,C
  015D1 59              LD      E,C
  015D2 CDA208          CALL    08A2H           ;X = BCDE / X = 1 / argument
  015D5 211007          LD      HL,0710H        ;HL = address of X = (HL) - X
  015D8 E5              PUSH    HL              ;Save as new RET address
  015D9 21E315          LD      HL,15E3H        ;HL -> coefficient table
  015DC CD9A14          CALL    149AH           ;Calculate series 1
  015DF 218B15          LD      HL,158BH        ;HL -> PI/2
  015E2 C9              RET                     ;Result = PI/2 - X if
                                                ;argument was > 1
                                                ;(Series calculation is only
                                                ;correct for -1 < X < 1)
; Coefficients for ATN

  015E3 09              DEFB    09H             ;9 coefficients

  015E4 4A              DEFB    4AH             ;2.86623E-3     approx. 1/349
  015E5 D7              DEFB    D7H
  015E6 3B              DEFB    3BH
  015E7 78              DEFB    78H

  015E8 02              DEFB    02H             ;-0.0161657     approx. -1/62
  015E9 6E              DEFB    6EH
  015EA 84              DEFB    84H
  015EB 7B              DEFB    7BH

  015EC FE              DEFB    FEH             ;0.0429096      approx. 1/23
  015ED C1              DEFB    C1H
  015EE 2F              DEFB    2FH
  015EF 7C              DEFB    7CH

  015F0 74              DEFB    74H             ;-0.0752896     approx. -1/11
```

```
  015F1 31              DEFB    31H
  015F2 9A              DEFB    9AH
  015F3 7D              DEFB    7DH

  015F4 84              DEFB    84H             ;0.106563       approx. 1/9
  015F5 3D              DEFB    3DH
  015F6 5A              DEFB    5AH
  015F7 7D              DEFB    7DH

  015F8 C8              DEFB    C8H             ;-0.142089      approx. -1/7
  015F9 7F              DEFB    7FH
  015FA 91              DEFB    91H
  015FB 7E              DEFB    7EH

  015FC E4              DEFB    E4H             ;0.199936       = 1/5
  015FD BB              DEFB    BBH
  015FE 4C              DEFB    4CH
  015FF 7E              DEFB    7EH

  01600 6C              DEFB    6CH             ;-0.333331      = 1/3
  01601 AA              DEFB    AAH
  01602 AA              DEFB    AAH
  01603 7F              DEFB    7FH

  01604 00              DEFB    00H             ;1.0            = 1/1
  01605 00              DEFB    00H
  01606 00              DEFB    00H
  01607 81              DEFB    81H


; Address table for Level II and Disk BASIC functions (tokens D7H to FAH)

  01608 8A09            DEFW    098AH           ;SNG
  0160A 370B            DEFW    0B37H           ;INT
  0160C 7709            DEFW    0977H           ;ABS
  0160E D427            DEFW    27D4H           ;FRE
  01610 EF2A            DEFW    2AEFH           ;INP
  01612 F527            DEFW    27F5H           ;POS
  01614 E713            DEFW    13E7H           ;SQR
  01616 C914            DEFW    14C9H           ;RND
  01618 0908            DEFW    0809H           ;LOG
  0161A 3914            DEFW    1439H           ;EXP
  0161C 4115            DEFW    1541H           ;COS
  0161E 4715            DEFW    1547H           ;SIN
  01620 A815            DEFW    15A8H           ;TAN
  01622 BD15            DEFW    15BDH           ;ATN
  01624 AA2C            DEFW    2CAAH           ;PEEK
  01626 5241            DEFW    4152H           ;CVI
  01628 5841            DEFW    4158H           ;CVS
  0162A 5E41            DEFW    415EH           ;CVD
  0162C 6141            DEFW    4161H           ;EOF
  0162E 6441            DEFW    4164H           ;LOC
  01630 6741            DEFW    4167H           ;LOF
  01632 6A41            DEFW    416AH           ;MKI$
  01634 6D41            DEFW    416DH           ;MKS$
  01636 7041            DEFW    4170H           ;MKD$
  01638 7F0A            DEFW    0A7FH           ;CINT
  0163A B10A            DEFW    0AB1H           ;CSNG
  0163C DB0A            DEFW    0ADBH           ;CDBL
  0163E 260B            DEFW    0B26H           ;FIX
  01640 032A            DEFW    2A03H           ;LEN
  01642 3628            DEFW    2836H           ;STR$
  01644 C52A            DEFW    2AC5H           ;VAL
  01646 0F2A            DEFW    2A0FH           ;ASC
  01648 1F2A            DEFW    2A1FH           ;CHR$
  01649 612A            DEFW    2A61H           ;LEFT$
  0164A 912A            DEFW    2A91H           ;RIGHT$
  0164E 9A2A            DEFW    2A9AH           ;MID$


; Table for keywords of Level II and Disk BASIC
```

```
01650 C54E44          DEFB    80H+'E','ND'            ;END
01653 C64F52          DEFB    80H+'F','OR'            ;FOR
01656 D245534554      DEFB    80H+'R','ESET'          ;RESET
0165B D34554          DEFB    80H+'S','ET'            ;SET
0165E C34C53          DEFB    80H+'C','LS'            ;CLS
01661 C34D44          DEFB    80H+'C','MD'            ;CMD
01664 D2414E444F4D    DEFB    80H+'R','ANDOM          ;RANDOM
0166A CE455854        DEFB    80H+'N','EXT'           ;NEXT
0166E C4415441        DEFB    80H+'D','ATA'           ;DATA
01672 C94E505554      DEFB    80H+'I','NPUT'          ;INPUT
01677 C4494D          DEFB    80H+'D','IM'            ;DIM
0167A D2454144        DEFB    80H+'R','EAD'           ;READ
0167E CC4554          DEFB    80H+'L','ET'            ;LET
01681 C74F544F        DEFB    80H+'G','OTO'           ;GOTO
01685 D2554E          DEFB    80H+'R','UN'            ;RUN
01688 C946            DEFB    80H+'I','F'             ;IF
0168A D24553544F5245  DEFB    80H+'R','ESTORE'        ;RESTORE
01691 C74F535542      DEFB    80H+'G','OSUB'          ;GOSUB
01696 D2455455524E    DEFB    80H+'R','ETURN'         ;RETURN
0169C D2454D          DEFB    80H+'R','EM'            ;REM
0169F D3544F50        DEFB    80H+'S','TOP'           ;STOP
016A3 C54C5345        DEFB    80H+'E','LSE'           ;ELSE
016A7 D4524F4E        DEFB    80H+'T','RON'           ;TRON
016AB D4524F4646      DEFB    80H+'T','ROFF'          ;TROFF
016B0 C44546535452    DEFB    80H+'D','EFSTR'         ;DEFSTR
016B6 C44546494E54    DEFB    80H+'D','EFINT'         ;DEFINT
016BC C44546534E47    DEFB    80H+'D','EFSNG'         ;DEFSNG
016C2 C4454644424C    DEFB    80H+'D','EFDBL'         ;DEFDBL
016C8 CC494E45        DEFB    80H+'L','INE'           ;LINE
016CC C5444954        DEFB    80H+'E','DIT'           ;EDIT
016D0 C552524F52      DEFB    80H+'E','RROR'          ;ERROR
016D5 D24553554D45    DEFB    80H+'R','ESUME'         ;RESUME
016DB CF5554          DEFB    80H+'O','UT'            ;OUT
016DE CF4E            DEFB    80H+'O','N'             ;ON
016E0 CF50454E        DEFB    80H+'O','PEN'           ;OPEN
016E4 C649454C44      DEFB    80H+'F','IELD'          ;FIELD
016E9 C74554          DEFB    80H+'G','ET'            ;GET
016EC D05554          DEFB    80H+'P','UT'            ;PUT
016EF C34C4F5345      DEFB    80H+'C','LOSE'          ;CLOSE
016F4 CC4F4144        DEFB    80H+'L','OAD'           ;LOAD
016F8 CD45524745      DEFB    80H+'M','ERGE'          ;MERGE
016FD CE414D45        DEFB    80H+'N','AME'           ;NAME
01701 CB494C4C        DEFB    80H+'K','ILL'           ;KILL
01705 CC534554        DEFB    80H+'L','SET'           ;LSET
01709 D2534554        DEFB    80H+'R','SET'           ;RSET
0170D D3415645        DEFB    80H+'S','AVE'           ;SAVE
01711 D3595354454D    DEFB    80H+'S','YSTEM'         ;SYSTEM
01717 CC5052494E54    DEFB    80H+'L','PRINT'         ;LPRINT
0171D C44546          DEFB    80H+'D','EF'            ;DEF
01720 D04F4B45        DEFB    80H+'P','OKE'           ;POKE
01724 D052494E54      DEFB    80H+'P','RINT'          ;PRINT
01729 C34F4E54        DEFB    80H+'C','ONT'           ;CONT
0172D CC495354        DEFB    80H+'L','IST'           ;LIST
01731 CC4C495354      DEFB    80H+'L','LIST'          ;LLIST
01736 C4454C455445    DEFB    80H+'D','ELETE'         ;DELETE
0173C C155544F        DEFB    80H+'A','UTO'           ;AUTO
01740 C34C454152      DEFB    80H+'C','LEAR'          ;CLEAR
01745 C34C4F4144      DEFB    80H+'C','LOAD'          ;CLOAD
0174A C353415645      DEFB    80H+'C','SAVE'          ;CSAVE
0174F CE4557          DEFB    80H+'N','EW'            ;NEW
01752 D4414228        DEFB    80H+'T','AB('           ;TAB(
01756 D44F            DEFB    80H+'T','0'             ;TO
01758 C64E            DEFB    80H+'F','N'             ;FN
0175A D553494E47      DEFB    80H+'U','SING'          ;USING
0175F D64152505452    DEFB    80H+'V','ARPTR'         ;VARPTR
01765 D55352          DEFB    80H+'U','SR'            ;USR
01768 C5524C          DEFB    80H+'E','RN'            ;ERN
0176B C55252          DEFB    80H+'E','RR'            ;ERR
0176E D35452494E4724  DEFB    80H+'S','TRING$'        ;STRING$
01775 C94E535452      DEFB    80H+'I','NSTR'          ;INSTR
```

```
  0177A C34845434B      DEFB    80H+'C','HECK'          ;CHECK
  0177F D4494D4524      DEFB    80H+'T','IME$'          ;TIME$
  01784 CD454D          DEFB    80H+'M','EM'            ;MEM
  01787 C94E4B455924    DEFB    80H+'I','NKEY$'         ;INKEY$
  0178D D448454E        DEFB    80H+'T','HEN'           ;THEN
  01791 CE4F54          DEFB    80H+'N','OT'            ;NOT
  01794 D3544550        DEFB    80H+'S','TEP'           ;STEP
  01798 AB              DEFB    80H+'+'                 ;+
  01799 AD              DEFB    80H+'-'                 ;-
  0179A AA              DEFB    80H+'*'                 ;*
  0179B AF              DEFB    80H+'/'                 ;/
  0179C DB              DEFB    80H+'['                 ;[
  0179D C14E44          DEFB    80H+'A','ND'            ;AND
  017A0 CF52            DEFB    80H+'O','R'             ;OR
  017A2 BE              DEFB    80H+'>'                 ;>
  017A3 BD              DEFB    80H+'='                 ;=
  017A4 BC              DEFB    80H+'<'                 ;<
  017A5 D3474E          DEFB    80H+'S','GN'            ;SNG
  017A8 C94E54          DEFB    80H+'I','NT'            ;INT
  017AB C14253          DEFB    80H+'A','BS'            ;ABS
  017AE C65245          DEFB    80H+'F','RE'            ;FRE
  017B1 C94E50          DEFB    80H+'I','NP'            ;INP
  017B4 D04F53          DEFB    80H+'P','OS'            ;POS
  017B7 D35152          DEFB    80H+'S','QR'            ;SQR
  017BA D24E44          DEFB    80H+'R','ND'            ;RND
  017BD CC4F47          DEFB    80H+'L','OG'            ;LOG
  017C0 C55850          DEFB    80H+'E','XP'            ;EXP
  017C3 C34F53          DEFB    80H+'C','OS'            ;COS
  017C6 D3494E          DEFB    80H+'S','IN'            ;SIN
  017C9 D4414E          DEFB    80H+'T','AN'            ;TAN
  017CC C1544E          DEFB    80H+'A','TN'            ;ATN
  017CF D045454B        DEFB    80H+'P','EEK'           ;PEEK
  017D3 C35649          DEFB    80H+'C','VI'            ;CVI
  017D6 C35653          DEFB    80H+'C','VS'            ;CVS
  017D9 C35644          DEFB    80H+'C','VD'            ;CVD
  017DC C54F46          DEFB    80H+'E','OF'            ;EOF
  017DF CC4F43          DEFB    80H+'L','OC'            ;LOC
  017E2 CC4F46          DEFB    80H+'L','OF'            ;LOF
  017E5 CD4B4924        DEFB    80H+'M','KI$'           ;MKI$
  017E9 CD4B5324        DEFB    80H+'M','KS$'           ;MKS$
  017ED CD4B4424        DEFB    80H+'M','KD$'           ;MKD$
  017F1 C3494E54        DEFB    80H+'C','INT'           ;CINT
  017F5 C3534E47        DEFB    80H+'C','SNG'           ;CSNG
  017F9 C344424C        DEFB    80H+'C','DBL'           ;CDBL
  017FD C64958          DEFB    80H+'F','IX'            ;FIX
  01800 CC454E          DEFB    80H+'L','EN'            ;LEN
  01803 D3545224        DEFB    80H+'S','TR$'           ;STR$
  01807 D6414C          DEFB    80H+'V','AL'            ;VAL
  0180A C15343          DEFB    80H+'A','SC'            ;ASC
  0180D C3485224        DEFB    80H+'C','HR$'           ;CHR$
  01811 CC45465424      DEFB    80H+'L','EFT$'          ;LEFT$
  01816 D24947485424    DEFB    80H+'R','IGHT$'         ;RIGHT$
  0181C CD494424        DEFB    80H+'M','ID$'           ;MID$
  01820 A7              DEFB    80H+27H                 ;'
  01821 80              DEFB    80H                     ;End of table


; Address table for Level II and Disk BASIC statements (tokens 80H to BBH)

  01822 AE1D            DEFW    1DAEH           ;END
  01824 A11C            DEFW    1CA1H           ;FOR
  01826 3801            DEFW    0138H           ;RESET
  01828 3501            DEFW    0135H           ;SET
  0182A C901            DEFW    01C9H           ;CLS
  0182C 7341            DEFW    4173H           ;CMD
  0182E D301            DEFW    01D3H           ;RANDOM
  01830 B622            DEFW    22B6H           ;NEXT
  01832 051F            DEFW    1F05H           ;DATA
  01834 9A21            DEFW    219AH           ;INPUT
  01836 0826            DEFW    2608H           ;DIM
  01838 EF21            DEFW    21EFH           ;READ
```

```
  0183A 211F             DEFW    1F21H           ;LET
  0183C C21E             DEFW    1EC2H           ;GOTO
  0183E A31E             DEFW    1EA3H           ;RUN
  01840 3920             DEFW    2039H           ;IF
  01842 911D             DEFW    1D91H           ;RESTORE
  01844 B11E             DEFW    1EB1H           ;GOSUB
  01846 DE1E             DEFW    1EDEH           ;RETURN
  01848 071F             DEFW    1F07H           ;REM
  0184A A91D             DEFW    1DA9H           ;STOP
  0184C 071F             DEFW    1F07H           ;ELSE
  0184E F71D             DEFW    1DF7H           ;TRON
  01850 F81D             DEFW    1DF8H           ;TROFF
  01852 001E             DEFW    1E00H           ;DEFSTR
  01854 031E             DEFW    1E03H           ;DEFINT
  01856 061E             DEFW    1E06H           ;DEFSNG
  01858 091E             DEFW    1E09H           ;DEFDBL
  0185A A341             DEFW    41A3H           ;LINE
  0185C 602E             DEFW    2E60H           ;EDIT
  0185E F41F             DEFW    1FF4H           ;ERROR
  01860 AF1F             DEFW    1FAFH           ;RESUME
  01862 FB2A             DEFW    2AFBH           ;OUT
  01864 6C1F             DEFW    1F6CH           ;ON
  01866 7941             DEFW    4179H           ;OPEN
  01868 7C41             DEFW    417CH           ;FIELD
  0186A 7F41             DEFW    417FH           ;GET
  0186C 8241             DEFW    4182H           ;PUT
  0186E 8541             DEFW    4185H           ;CLOSE
  01870 8841             DEFW    4188H           ;LOAD
  01872 8B41             DEFW    418BH           ;MERGE
  01874 8E41             DEFW    418EH           ;NAME
  01876 9141             DEFW    4191H           ;KILL
  01878 9741             DEFW    4197H           ;LSET
  0187A 9A41             DEFW    419AH           ;RSET
  0187C A041             DEFW    41A0H           ;SAVE
  0187E B202             DEFW    02B2H           ;SYSTEM
  01880 6720             DEFW    2067H           ;LPRINT
  01882 5B41             DEFW    415BH           ;DEF
  01884 B12C             DEFW    2CB1H           ;POKE
  01886 6F20             DEFW    206FH           ;PRINT
  01888 E41D             DEFW    1DE4H           ;CONT
  0188A 2E2B             DEFW    2B2EH           ;LIST
  0188C 292B             DEFW    2B29H           ;LLIST
  0188E C62B             DEFW    2BC6H           ;DELETE
  01890 0820             DEFW    2008H           ;AUTO
  01892 7A1E             DEFW    1E7AH           ;CLEAR
  01894 1F2C             DEFW    2C1FH           ;CLOAD
  01896 F52B             DEFW    2BF5H           ;CSAVE
  01898 491B             DEFW    1B49H           ;NEW


; Prioritytable for operators
; highest value corresponds with highest proirity

  0189A 79               DEFB    79H             ;+
  0189B 79               DEFB    79H             ;-
  0189C 7C               DEFB    7CH             ;*
  0189D 7C               DEFB    7CH             ;/
  0189E 7F               DEFB    7FH             ;Exponent
  0189F 50               DEFB    50H             ;AND
  018A0 46               DEFB    46H             ;OR


; Address table for type conversion

  018A1 DB0A             DEFW    0ADBH           ;0ADBH: CDBL
  018A3 0000             DEFW    0000H           ;0000H: unused entry in table
  018A5 7F0A             DEFW    0A7FH           ;0A7FH: CINT
  018A7 F40A             DEFW    0AF4H           ;0AF4H: ?TM error when no
                                                 ;       string in X
  018A9 B10A             DEFW    0AB1H           ;0AB1H: CSNG
```

```
; Address table for basic arithmatic and comparisons

; 1. double precision

  018AB 770C            DEFW    0C77            ; + (X = X + Y)
  018AD 700C            DEFW    0C70            ; - (X = X - Y)
  018AF A10D            DEFW    0DA1            ; * (X = X * Y)
  018B1 E50D            DEFW    0DE5            ; / (X = X / Y)
  018B3 780A            DEFW    0A78            ; Compare (CP X , Y)

; 2. single precision

  018B5 1607            DEFW    0716            ; + (X = BCDE + X)
  018B7 1307            DEFW    0713            ; - (X = BCDE - X)
  018B9 4708            DEFW    0847            ; * (X = BCDE * X)
  018BB A208            DEFW    08A2            ; / (X = BCDE / X)
  018BD 0C0A            DEFW    0A0C            ; Compare (CP X , BCDE)

; 3. integer

  018BF D20B            DEFW    0BD2            ; + (X = DE + HL)
  018C1 C70B            DEFW    0BC7            ; - (X = DE - HL)
  018C3 F20B            DEFW    0BF2            ; * (X = DE * HL)
  018C5 9024            DEFW    2490            ; / (X = DE / HL)
  018C7 390A            DEFW    0A39            ; Compare (CP HL , DE)


; Table with character combinations for error messages

  018C9 4E46            DEFB    'NF'
  018CB 534E            DEFB    'SN'
  018CD 5247            DEFB    'RG'
  018CF 4F44            DEFB    'OD'
  018D1 4643            DEFB    'FC'
  018D3 4F46            DEFB    'OF'
  018D5 4F4D            DEFB    'OM'
  018D7 554C            DEFB    'UL'
  018D9 4253            DEFB    'BS'
  018DB 4444            DEFB    'DD'
  018DD 2F30            DEFB    '/0'
  018DF 4944            DEFB    'ID'
  018E1 544D            DEFB    'TM'
  018E3 4F53            DEFB    'OS'
  018E5 4C53            DEFB    'LS'
  018E7 5354            DEFB    'ST'
  018E9 434E            DEFB    'CN'
  018EB 4E52            DEFB    'NR'
  018ED 5257            DEFB    'RW'
  018EF 5545            DEFB    'UE'
  018F1 4D4F            DEFB    'MO'
  018F3 4644            DEFB    'FD'
  018F5 534E            DEFB    'SN'


; Following 39 bytes are copied into system RAM from 4080H onwards

  018F7 D600            SUB     00H
  018F9 6F              LD      L,A
  018FA 7C              LD      A,H
  018FB DE00            SBC     A,00H
  018FD 67              LD      H,A
  018FE 78              LD      A,B
  018FF DE00            SBC     A,00H
  01901 47              LD      B,A
  01902 3E00            LD      A,00H
  01904 C9              RET
  01905 4A              LD      C,D
  01906 1E40            LD      E,40H
  01908 E64D            AND     4DH
  0190A DB00            IN      A,(00H)
```

```
  0190C C9                RET
  0190D D300              OUT     (00H),A
  0190F C9                RET

  01910 00                DEFB    00H
  01911 00                DEFB    00H
  01912 00                DEFB    00H
  01913 00                DEFB    00H
  01914 28                DEFB    28H
  01914 1E                DEFB    1EH
  01916 00                DEFB    00H
  01917 4C                DEFB    4CH
  01918 43                DEFB    43H
  01919 FE                DEFB    FEH
  01919 FF                DEFB    FFH
  0191B 01                DEFB    01H
  0191C 48                DEFB    48H


; Text ' Error'

  0191D 204572726F72      DEFB    ' Error'
  01923 00                DEFB    00H             ;End of string

; Text ' in '

  01924 2069              DEFB    ' in '
  01928 00                DEFB    00H             ;End of string

; Text 'READY'

  01929 5245414459        DEFB    'READY'
  0192E 0D                DEFB    0DH             ;Carriage Return
  0192F 00                DEFB    00H             ;End of string

; Text 'Break'

  01930 427265616B        DEFB    'Break'
  01935 00                DEFB    00H             ;End of string


; SUB for FOR, NEXT and RETURN
; Retrieves data from stack
;
; I: DE = VARPTR of new loop variable when a new FOR-TO loop is started
;    DE = VARPTR of the variable indicated with NEXT
;    DE = 0000H when NEXT with no variable
; O: DE = unchanged
;    HL = 'stackpointer' on FOR-TO-stack + 1 (when Z-flag = 0)
;    HL = 'stackpointer' on FOR-TO-stack + 3 (when Z-flag = 1)
;    Z-flag = 0 when no FOR-TO-stack found or when the variable is not used
;               in a loop
;    Z-flag = 1 when call from NEXT or if the variable had already been used
;               in a loop

  01936 210400            LD      HL,0004H
  01939 39                ADD     HL,SP           ;HL = SP + 4
                                                  ;HL is now like stack pointer
                                                  ;after 2 POPs
  0193A 7E                LD      A,(HL)          ;Get marker form stack
  0193B 23                INC     HL              ;"stack pointer" + 1
  0193C FE81              CP      81H             ;FOR marker found ?
  0193E C0                RET     NZ              ;No: done, return
                                                  ;Yes:
  0193F 4E                LD      C,(HL)          ;BC = VARPTR of loop variable
  01940 23                INC     HL
  01941 46                LD      B,(HL)
  01942 23                INC     HL
  01943 E5                PUSH    HL              ;save "stack pointer"
  01944 69                LD      L,C             ;HL -> loop variable
  01945 60                LD      H,B
```

```
  01946 7A              LD      A,D             ;DE = 0000H (call from NEXT) ?
  01947 B3              OR      E
  01948 EB              EX      DE,HL           ;DE = VARPTR of loop variable
                                                ;found in stack
                                                ;HL = VARPTR of new loop
                                                ;variable
                                                ;Call from NEXT ?
  01949 2802            JR      Z,194DH         ;Yes: done

  0194B EB              EX      DE,HL           ;Swap DE and HL
  0194C DF              RST     18H             ;Compare both VARPTRs
  0194D 010E00          LD      BC,000EH        ;BC = offset to next
                                                ;FOR-TO-stack
  01950 E1              POP     HL              ;Restore "stack pointer"
  01951 C8              RET     Z               ;Done if searched VARPTR found

  01952 09              ADD     HL,BC           ;Increment "stack pointer" to
                                                ;next FOR-TO-stack.
                                                ;(Every FOR-TO-loop requires
                                                ;17 bytes of stack space. HL
                                                ;was already incremented by 3
                                                ;(193BH, 1940H, 1942H). Add
                                                ;0EH (14) to get 17)
  01953 18E5            JR      193AH           ;Continue search in stack


; Move program text for insertion of a new line
; Copies memory from (BC) to (HL) until BC = DE
;
; I: BC -> old program end (before move)
;    DE = LP of new line
;    HL -> new program end (after move)
; O: DE = LP on new line
;    HL = DE

  01955 CD6C19          CALL    196CH           ;Enough free memory available ?
  01958 C5              PUSH    BC              ;Swap BC and HL
  01959 E3              EX      (SP),HL         ;(because of RST 18H)
  0195A C1              POP     BC
  0195B DF              RST     18H             ;New line reached ?
  0195C 7E              LD      A,(HL)          ;Copy byte from old location
  0195D 02              LD      (BC),A          ;to new location
  0195E C8              RET     Z               ;Yes: done, return

  0195F 0B              DEC     BC              ;Pointer - 1
  01960 2B              DEC     HL
  01961 18F8            JR      195BH           ;Copy next byte


; Test if sufficient memory space is available
; ?OM Error if less than 2 * C bytes are available
;
; I: C = number of required bytes

  01963 E5              PUSH    HL              ;Save PTP
  01964 2AFD40          LD      HL,(40FDH)      ;HL -> start of free memory
  01967 0600            LD      B,00H           ;BC = number of required bytes
  01969 09              ADD     HL,BC           ;HL - HL + 2 * BC
  0196A 09              ADD     HL,BC
  0196B 3EE5            LD      A,0E5H          ;--

; Is there sufficient free space from (HL) onwards ?
;
; I: HL -> free memory space

  0196B   E5            PUSH    HL              ;Save HL
                                                ;HL now points to the new
                                                ;start of free memory
  0196D 3EC6            LD      A,0C6H          ;HL = FFC6H - HL
  0196F 95              SUB     L
  01970 6F              LD      L,A
```

```
  01971 3EFF             LD      A,0FFH
  01973 9C               SBC     A,H
                                                ;HL > FFC6H
  01974 3804             JR      C,197AH        ;Yes: ?OM Error

  01976 67               LD      H,A            ;MSB back to H
  01977 39               ADD     HL,SP          ;HL = SP + (FFC6 - HL)
  01978 E1               POP     HL             ;Restore HL
  01979 D8               RET     C              ;If room until stack then
                                                ;return else ?OM Error
  0197A 1E0C             LD      E,0CH          ;E = error code for ?OM Error
  0197C 1824             JR      19A2H          ;Continue at error routine


; End program without 'END'

  0197E 2AA240           LD      HL,(40A2H)     ;HL = current LN
  01981 7C               LD      A,H            ;Was a program finished ?
  01982 A5               AND     L              ;(is LN <> 65535)
  01983 3C               INC     A
  01984 2808             JR      Z,198EH        ;No: go via 198EH to END

  01986 3AF240           LD      A,(40F2H)      ;ON ERROR GOTO flag set ?
  01989 B7               OR      A

  0198A 1E22             LD      E,22H          ;E = error code for ?NR error
  0198C 2014             JR      NZ,19A2H       ;Flag set: ?NR Error

  0198E C3C11D           JP      1DC1H          ;Continue at END


; ?SN Error in DATA line

  01991 2ADA40           LD      HL,(40DAH)     ;HL = DATA LN
  01994 22A240           LD      (40A2H),HL     ;Store as current LN

; ?SN Error

  01997 1E02             LD      E,02H          ;E = error code for ?SN Error
  01999 011E14           LD      BC,141EH       ;--

; ?/0 Error

* 0199A   1E14           LD      E,14H          ;E = error code for ?/0 Error
  0199C 011E00           LD      BC,001EH       ;--

; ?NF Error
* 0199D   1E00           LD      E,00H          ;E = error code for ?NF Error
  0199F 011E24           LD      BC,241EH       ;--

; ?RW Error

* 019A0   1E24           LD      E,24H          ;E = error code for ?RW Error


; Error routine
; Displays the error code and line number and aborts program
;
; I: E = (error code - 1) * 2
; O: - (returns to active command mode or to error handling routine if
;    ON ERROR GOTO was active

  019A2 2AA240           LD      HL,(40A2H)     ;HL = current LN
  019A5 22EA40           LD      (40EAH),HL     ;Save as ERL
  019A8 22EC40           LD      (40ECH),HL     ;and '.' (current LN)
  019AB 01B419           LD      BC,19B4H       ;19B4H = new return address
  019AE 2AE840           LD      HL,(40E8H)     ;HL = last SP value
  019B1 C39A1B           JP      1B9AH          ;Reinitialize stack

  019B4 C1               POP     BC             ;Correct stack
  019B5 7B               LD      A,E            ;A = error code
```

```
  019B6 4B              LD      C,E             ;C = error code
  019B7 329A40          LD      (409AH),A       ;Save error code as ERR
  019BA 2AE640          LD      HL,(40E6H)      ;HL = PTP before error
  019BD 22EE40          LD      (40EEH),HL      ;Save PTP for RESUME
  019C0 EB              EX      DE,HL           ;DE = PTP
  019C1 2AEA40          LD      HL,(40EAH)      ;HL = LN
  019C4 7C              LD      A,H             ;LN = FFFH (65535 ?)
  019C5 A5              AND     L
  019C6 3C              INC     A
  019C7 2807            JR      Z,19D0H         ;Yes: error on active command
                                                ;mode level, continue at 19D0H
  019C9 22F540          LD      (40F5H),HL      ;Save LN for CONT
  019CC EB              EX      DE,HL           ;HL = PTP
  019CD 22F740          LD      (40F7H),HL      ;Save PTP for CONT
  019D0 2AF040          LD      HL,(40F0H)      ;HL = LN of ON ERROR GOTO
  019D3 7C              LD      A,H             ;ON ERROR GOTO active ?
  019D4 B5              OR      L               ;(LN <> 0 ?)
  019D5 EB              EX      DE,HL           ;DE = LN
  019D6 21F240          LD      HL,40F2H        ;HL = ON ERROR GOT flag
  019D9 2808            JR      Z,19E3H         ;No: continue at 19E3H

  019DB A6              AND     (HL)            ;ON ERROR GOTO Flag set ?
  019DC 2005            JR      NZ,19E3H        ;No: continue at 19E3H

  019DE 35              DEC     (HL)            ;Flag - 1 (as counter
                                                ;for RESUME)
  019DF EB              EX      DE,HL           ;HL = PTP of ON ERROR GOTO line
  019E0 C3361D          JP      1D36H           ;Continue program at (HL)

; Issue error

  019E3 AF              XOR     A               ;Clear ON ERROR GOTO flag
  019E4 77              LD      (HL),A
  019E5 59              LD      E,C             ;E = Error code
  019E6 CDF920          CALL    20F9H           ;Start at new line
  019E9 21C918          LD      HL,18C9H        ;HL -> Error codes table
  019EC CDA641          CALL    41A6H           ;DOS
  019EF 57              LD      D,A             ;DE = offset in error table
  019F0 3E3F            LD      A,3FH           ;A = '?'
  019F2 CD2A03          CALL    032AH           ;Print it
  019F5 19              ADD     HL,DE           ;HL -> Error code text
  019F6 7E              LD      A,(HL)          ;Print both characters
  019F7 CD2A03          CALL    032AH           ;Print 1st error character
  019FA D7              RST     10H             ;A = 2nd character
  019FB CD2A03          CALL    032AH           ;Print 2nd error character
  019FE 211D19          LD      HL,191DH        ;HL -> ' Error'
  01A01 E5              PUSH    HL              ;Save HL
  01A02 2AEA40          LD      HL,(40EAH)      ;HL = ERL
  01A05 E3              EX      (SP),HL         ;Save ERL and restore text
                                                ;pointer
; Entry for STOP (see 1DDEH)

  01A06 CD7935          CALL    3579H           ;Give text and sound
  01A09 E1              POP     HL              ;Restore ERL
  01A0A 11FEFF          LD      DE,0FFFEH       ;Error originates from
                                                ;MEM SIZE?
                                                ;(ERL = 65534)
  01A0D DF              RST     18H             ;Compare HL and DE
  01A0E CA7406          JP      Z,0674H         ;Yes: back to start 1

  01A11 7C              LD      A,H             ;Error originates from program?
  01A12 A5              AND     L               ;(ERL <> 65535)
  01A13 3C              INC     A
  01A14 C4A70F          CALL    NZ,0FA7H        ;Yes: print ' in ' and LN
  01A17 3EC1            LD      A,0C1H          ;Back into BASIC
* 01A18   C1            POP     BC              ;--

; Entry into active command mode

  01A19 CD8B03          CALL    038BH           ;End output to printer,
                                                ;next output to screen
```

```
  01A1C CDAC41          CALL    41ACH           ;DOS
  01A1F 00              NOP                     ;--
  01A20 00              NOP
  01A21 00              NOP
  01A22 CDF920          CALL    20F9H           ;Finish screen output
  01A25 212919          LD      HL,1929H        ;HL -> READY text
  01A28 CD9238          CALL    3892H           ;Set LGR, NBGRD and print text
  01A2B 3A9A40          LD      A,(409AH)       ;A = last error code
  01A2E D602            SUB     02H             ;Was it ?SN Error ?
  01A30 CC532E          CALL    Z,2E53H         ;Yes: call EDIT

  01A33 21FFFF          LD      HL,0FFFFH       ;Set current LN to 65535
  01A36 22A240          LD      (40A2H),HL
  01A39 3AE140          LD      A,(40E1H)       ;AUTO active ?
  01A3C B7              OR      A
  01A3D 2837            JR      Z,1A76H         ;No: goto normal command mode

; Process AUTO

  01A3F 2AE240          LD      HL,(40E2H)      ;HL = AUTO-LN
  01A42 E5              PUSH    HL              ;Save LN
  01A43 CDAF0F          CALL    0FAFH           ;Print LN
  01A46 D1              POP     DE              ;DE = LN
  01A47 D5              PUSH    DE              ;Save LN
  01A48 CD2C1B          CALL    1B2CH           ;Search LN and test if already
                                                :present
  01A4B 3E2A            LD      A,'*'           ;A = '*'
                                                ;LN already present ?
  01A4D 3802            JR      C,1A51H         ;Yes: continue at 1A51H

  01A4F 3E20            LD      A,20H           ;A = ' '

  01A51 CD2A03          CALL    032AH           ;Print it
  01A54 CD6103          CALL    0361H           ;Input line
  01A57 D1              POP     DE              ;Restore LN
                                                ;<BREAK> pressed ?
  01A58 3006            JR      NC,1A60H        ;No: accept line

  01A5A AF              XOR     A               ;A = 00H
  01A5B 32E140          LD      (40E1H),A       ;Switch off AUTO
  01A5E 18B9            JR      1A19H           ;Back to active command mode


; Accept AUTO line

  01A60 2AE440          LD      HL,(40E4H)      ;HL = diffence to next LN
  01A63 19              ADD     HL,DE           ;Calculate next AUTO-LN
  01A64 38F4            JR      C,1A5AH         ;Abort AUTO if next LN
                                                ;becomes larger than 65535
  01A66 D5              PUSH    DE              ;Save current LN
  01A67 11F9FF          LD      DE,0FFF9H       ;DE = 65530 (maximum LN + 1)
  01A6A DF              RST     18H             ;Compare new LN with 65530
  01A6B D1              POP     DE              ;Restore current LN
                                                ;new LN too large (> 65529) ?
  01A6C 30EC            JR      NC,1A5AH        ;Yes: abort AUTO

  01A6E 22E240          LD      (40E2H),HL      ;Store next LN
  01A71 F6FF            OR      0FFH            ;Set A <> 0
  01A73 C3EB2F          JP      2FEBH           ;Accept line by EDIT routine


; Normal command or line input in active command mode

  01A76 3E3E            LD      A,3EH           ;A = '>'
  01A78 CD2A03          CALL    032AH           ;Print it
  01A7B CD6103          CALL    0361H           ;Input line
                                                ;<BREAK> pressed ?
  01A7E DA331A          JP      C,1A33H         ;Yes: back to active
                                                ;command mode
  01A81 D7              RST     10H             ;A = 1st character of
                                                ;entered line
```

```
  01A82 3C              INC     A               ;A = 00H ?
  01A83 3D              DEC     A               ;(no 'OR A' so that C-flag is
                                                ;not infuenced)
  01A84 CA331A          JP      Z,1A33H         ;Yes: back to active command
                                                ;mode
  01A87 F5              PUSH    AF              ;Save flags
  01A88 CD5A1E          CALL    1E5AH           ;Decode number
  01A8B 2B              DEC     HL              ;Set HL back to the
  01A8C 7E              LD      A,(HL)          ;All following spaces are
  01A8D FE20            CP      20H             ;last digit of the number
  01A8F 28FA            JR      Z,1A8BH

  01A91 23              INC     HL              ;HL -> first character in line
  01A92 7E              LD      A,(HL)          ;A = character
  01A93 FE20            CP      20H             ;Space ?
  01A95 CCC909          CALL    Z,09C9H         ;Yes: HL + 1, the first space
                                                ;is not used because LIST
                                                ;automaticcaly inserts a space
                                                ;after the line number
  01A98 D5              PUSH    DE              ;Save LN
  01A99 CDC01B          CALL    1BC0H           ;Tokenize line
  01A9C D1              POP     DE              ;Restore LN
  01A9D F1              POP     AF              ;Restore flags
  01A9E 22E640          LD      (40E6H),HL      ;Save HL as current PTP
  01AA1 CDB241          CALL    41B2H           ;DOS
                                                ;LN indicated ?
  01AA4 D25A1D          JP      NC,1D5AH        ;No: execute line directly

; Use line to program

  01AA7 D5              PUSH    DE              ;Save LN
  01AA8 C5              PUSH    BC              ;Save length of line
  01AA9 AF              XOR     A               ;A = 00H
  01AAA 32DD40          LD      (40DDH),A       ;Clear STOP-flag
  01AAD D7              RST     10H             ;A = 1st character of tokenized
                                                ;line
  01AAE B7              OR      A               ;A = 00H ?
                                                ;(No program text entered
                                                ;behind the line number ? )
  01AAF F5              PUSH    AF              ;Save flags
  01AB0 EB              EX      DE,HL           ;Save pointer
  01AB1 22EC40          LD      (40ECH),HL      ;Save LN as '.'-LN
  01AB4 EB              EX      DE,HL           ;Restore pointer
  01AB5 CD2C1B          CALL    1B2CH           ;LN already used in program ?
  01AB8 C5              PUSH    BC              ;Save LP
  01AB9 DCE42B          CALL    C,2BE4H         ;Yes: delete line
  01ABC D1              POP     DE              ;Restore LP
  01ABD F1              POP     AF              ;Restore flags
  01ABE D5              PUSH    DE              ;Save LP
                                                ;Z-flag = 1 (see 1AAEH) ?
  01ABF 2827            JR      Z,1AE8H         ;Yes: the line had only to be
                                                ;deleted. Now only the LPs in
                                                ;the program have to be renewed
  01AC1 D1              POP     DE              ;Restore LP
  01AC2 2AF940          LD      HL,(40F9H)      ;HL -> program end
  01AC5 E3              EX      (SP),HL         ;HL = length of line
  01AC6 C1              POP     BC              ;BC -> program end
  01AC7 09              ADD     HL,BC           ;HL -> new program end
  01AC8 E5              PUSH    HL              ;Save pointer
  01AC9 CD5519          CALL    1955H           ;Make room for new line
  01ACC E1              POP     HL              ;Restore pointer to new program
  01ACD 22F940          LD      (40F9H),HL      ;end and store it in system RAM
  01AD0 EB              EX      DE,HL           ;HL = new LP
  01AD1 74              LD      (HL),H          ;Set LP to next line <> 0
  01AD2 D1              POP     DE              ;Restore LN
  01AD3 E5              PUSH    HL              ;Save LP
  01AD4 23              INC     HL              ;LP + 2
  01AD5 23              INC     HL
  01AD6 73              LD      (HL),E          ;Insert new LP in program
  01AD7 23              INC     HL
  01AD8 72              LD      (HL),D
```

```
  01AD9 23              INC     HL
  01ADA EB              EX      DE,HL           ;DE -> free space for line text
  01ADB 2AA740          LD      HL,(40A7H)      ;HL -> line buffer
  01ADE EB              EX      DE,HL           ;HL -> program
                                                ;DE -> new line + 2
  01ADF 1B              DEC     DE              ;DE - 2
  01AE0 1B              DEC     DE

  01AE1 1A              LD      A,(DE)          ;copy byte from buffer
  01AE2 77              LD      (HL),A          ;into program
  01AE3 23              INC     HL              ;Pointer + 1
  01AE4 13              INC     DE
  01AE5 B7              OR      A               ;End of line reached ?
  01AE6 20F9            JR      NZ,1AE1H        ;No: continue copy
                                                ;Yes:
  01AE8 D1              POP     DE              ;DE = LP of new line
  01AE9 CDFC1A          CALL    1AFCH           ;Renew all LPs in program
                                                ;from DE onwards
  01AEC CDB541          CALL    41B5H           ;DOS
  01AEF CD5D1B          CALL    1B5DH           ;CLEAR
  01AF2 CDB841          CALL    41B8H           ;DOS
  01AF5 C3331A          JP      1A33H           ;Back to active command mode

; Renew all LPs in program

  01AF8 2AA440          LD      HL,(40A4H)      ;HL = start of program
  01AFB EB              EX      DE,HL           ;DE = start of program

; Renew all LPs in program from DE onwards

  01AFC 62              LD      H,D             ;HL = LP
  01AFD 6B              LD      L,E
  01AFE 7E              LD      A,(HL)          ;LP to next line = 0 ?
  01AFF 23              INC     HL
  01B00 B6              OR      (HL)            ;(end of program reached ?)
  01B01 C8              RET     Z               ;Yes: done, return
                                                ;No:
  01B02 23              INC     HL              ;Increment HL to line text
  01B03 23              INC     HL
  01B04 23              INC     HL
  01B05 AF              XOR     A               ;A = 00H

  01B06 BE              CP      (HL)            ;Search for 00H (end of line)
  01B07 23              INC     HL              ;Pointer + 1
  01B08 20FC            JR      NZ,1B06H        ;Continue search

  01B0A EB              EX      DE,HL           ;DE = LP on next line
  01B0B 73              LD      (HL),E          ;Set line
  01B0C 23              INC     HL
  01B0D 72              LD      (HL),D
  01B0E 18EC            JR      1AFCH           ;Process next line


; Decode line numbers at LIST and DELETE
; (LIST. , LIST FF , LIST FF-LL , LIST FF- , LIST -LL etc. )
;
; I: HL = PTP on character following command (line number in ASCII format)
; O: BC -> first line (FF in example)
;    DE = starting LN (FF in example) (default = 0)
;    HL -> next line (after FF)
;    (SP) = end-LN (LL in example) (default = 65529)

  01B10 110000          LD      DE,0000H        ;Default first LN = 0
  01B13 D5              PUSH    DE              ;Save first LN
                                                ;LNs indicated ?
  01B14 2809            JR      Z,1B1FH         ;No: continue at 1B1FH
                                                ;Yes:
  01B16 D1              POP     DE              ;Restore first LN
  01B17 CD4F1E          CALL    1E4FH           ;Decode first LN
  01B1A D5              PUSH    DE              ;Save first LN
                                                ;( = 0 if '-' indicated)
```

```
                                                      ;Only 1 LN indicated ?
  01B1B 280B              JR      Z,1B28H          ;Yes: continue at 1B1FH

  01B1D CF                RST     08H              ;'-' indicated ?
  01B1E CE                DEFB    0CEH             ;'-' token
  01B1F 11FAFF            LD      DE,FFFAH         ;Default last LN = 65529
  01B22 C44F1E            JP      Z,1E4FH          ;Decode last LN
  01B25 C29719            JR      NZ,1997          ;?SN Error ?
  01B28 EB                EX      DE,HL            ;HL = last LN
  01B29 D1                POP     DE               ;Restore first LN
  01B2A E3                EX      (SP),HL          ;Save last LN
  01B2B E5                PUSH    HL               ;RET address back on stack


; Search line number DE in program
;
; I: DE = line number of line to be searched
; O: BC -> searched line (if line found) or program end (line not found)
;    DE = line number
;    HL -> next line
;    Z-flag = 1, C-flag = 1: search succesfull
;    Z-flag = 1, C-flag = 0: BC -> program end
;    Z-flag = 0, C-flag = 0: BC -> line with largerst LN < searched LN

  01B2C 2AA440            LD      HL,(40A4H)       ;HL -> first program lline
  01B2F 44                LD      B,H              ;BC = LP
  01B30 4D                LD      C,L
  01B31 7E                LD      A,(HL)           ;LP to next line = 0 ?
  01B32 23                INC     HL
  01B33 B6                OR      (HL)
  01B34 2B                DEC     HL               ;Reverse INC HL
  01B35 C8                RET     Z                ;Yes: done, return

  01B36 23                INC     HL               ;Increment HL to line number
  01B37 23                INC     HL
  01B38 7E                LD      A,(HL)           ;HL = line number
  01B39 23                INC     HL
  01B3A 66                LD      H,(HL)
  01B3B 6F                LD      L,A
  01B3C DF                RST     18H              ;Compare with line number
                                                   ;to be searched for
  01B3D 60                LD      H,B              ;LP back to HL
  01B3E 69                LD      L,C
  01B3F 7E                LD      A,(HL)           ;HL -> next line
  01B40 23                INC     HL
  01B41 66                LD      H,(HL)
  01B42 6F                LD      L,A
  01B43 3F                CCF                      ;C-flag = 1 for succesfull
                                                   ;search
  01B44 C8                RET     Z                ;Done if line found

  01B45 3F                CCF                      ;Reverse C-flag again
  01B46 D0                RET     NC               ;RET if LN found >
                                                   ;LN to be searched for
  01B47 18E6              JR      1B2FH            ;Check next line


; NEW statement
; -------------

  01B49 C0                RET     NZ               ;?SN Error ?

  01B4A CDC901            CALL    01C9H            ;CLS
  01B4D 2AA440            LD      HL,(40A4H)       ;HL -> start of BASIC program
  01B50 CDF81D            CALL    1DF8H            ;TROFF
  01B53 32E140            LD      (40E1H),A        ;Switch off AUTO
  01B56 77                LD      (HL),A           ;Set LP to second line
  01B57 23                INC     HL               ;to zero
  01B58 77                LD      (HL),A
  01B59 23                INC     HL
  01B5A 22F940            LD      (40F9H),HL       ;Program end = program start+2
```

```
; Entry for RUN without line number (RET on program loop)

  01B5D 2AA440          LD      HL,(40A4H)      ;HL -> BASIC program start
  01B60 2B              DEC     HL              ;HL - 1


; CLEAR without argument

  01B61 22DF40          LD      (40DFH),HL      ;Store PTP to first/next
                                                ;command
  01B64 061A            LD      B,1AH           ;DEFSNG A-Z (B = 26)
  01B66 210141          LD      HL,4101H        ;HL -> DEF table
  01B69 3604            LD      (HL),04H        ;Set type code on SNG
  01B6B 23              INC     HL              ;Pointer + 1
  01B6C 10FB            DJNZ    1B69H           ;Loop

  01B6E AF              XOR     A               ;A = 00H
  01B6F 32F240          LD      (40F2H),A       ;Clear ON ERROR GOTO flag
  01B72 6F              LD      L,A             ;HL = 0000H
  01B73 67              LD      H,A
  01B74 22F040          LD      (40F0H),HL      ;ON ERROR GOTO LN = 0
  01B77 22F740          LD      (40F7H),HL      ;CONT PTP = 0
  01B7A 2AB140          LD      HL,(40B1H)      ;HL = TOPMEM
  01B7D 22D640          LD      (40D6H),HL      ;Address of last string in
                                                ;string space = TOPMEM
                                                ;(delete all strings)
  01B80 CD911D          CALL    1D91H           ;RESTORE
  01B83 2AF940          LD      HL,(40F9H)      ;HL -> BASIC program end
  01B86 22FB40          LD      (40FBH),HL      ;End of variable tables
  01B89 22FD40          LD      (40FDH),HL      ;is end of BASIC program:
                                                ;Delete all variables
  01B8C CDBB41          CALL    41BBH           ;DOS
  01B8F C1              POP     BC              ;BC = return address
  01B90 2AA040          LD      HL,(40A0H)      ;HL -> start of string memory
  01B93 2B              DEC     HL              ;HL - 2
  01B94 2B              DEC     HL
  01B95 22E840          LD      (40E8H),HL      ;Set new program stack
  01B98 23              INC     HL
  01B99 23              INC     HL

  01B9A F9              LD      SP,HL           ;SP = start of string memory
  01B9B 21B540          LD      HL,40B5H        ;HL -> start of string table
  01B9E 22B340          LD      (40B3H),HL      ;Set pointer to first free
                                                ;position in string table
  01BA1 CD8B03          CALL    038BH           ;End output to printer
  01BA4 CD6921          CALL    2169H           ;Next output to screen
  01BA7 AF              XOR     A               ;A = 0
  01BA8 67              LD      H,A             ;HL = 0000H
  01BA9 6F              LD      L,A
  01BAA 32DC40          LD      (40DCH),A       ;Release array variables
  01BAD E5              PUSH    HL              ;Put HL on stack
  01BAE C5              PUSH    BC              ;Set new return address
  01BAF 2ADF40          LD      HL,(40DFH)      ;HL = entry address SYSTEM
  01BB2 C9              RET


; SUB for INPUT
; Print '? ' and input line

  01BB3 3E3F            LD      A,3FH           ;A = '?'
  01BB5 CD2A03          CALL    032AH           ;Print it
  01BB8 3E20            LD      A,20H           ;A = ' '
  01BBA CD2A03          CALL    032AH           ;Print it
  01BBD C36103          JP      0361H           ;Input line


; SUB for active command mode
; Tokenize program text in line buffer
; Text pointer = pointer on entered text
```

```
; Buffer pointer = pointer on tokenized text
;
; I: HL = text pointer
; O: BC = number of required memory bytes (line length)
;    DE -> end of tokenized text
;    HL -> start of tokenized text - 1

  01BC0 AF              XOR     A               ;A = 00H
  01BC1 32B040          LD      (40B0H),A       ;Allow tokenizing
  01BC4 4F              LD      C,A             ;Counter = 0
  01BC5 EB              EX      DE,HL           ;DE = text pointer
  01BC6 2AA740          LD      HL,(40A7H)      ;HL -> start of line buffer
  01BC9 2B              DEC     HL
  01BCA 2B              DEC     HL
  01BCB EB              EX      DE,HL           ;DE = buffer pointer
                                                ;HL = text pointer
  01BCC 7E              LD      A,(HL)          ;A = text character
  01BCD FE20            CP      20H             ;Space ?
  01BCF CA5B1C          JP      Z,1C5BH         ;Yes: store character
                                                ;No:
  01BD2 47              LD      B,A             ;B = character
  01BD3 FE22            CP      22H             ;Start of a string ?
  01BD5 CA771C          JP      Z,1C77H         ;Yes: continue at 1C77H
                                                ;No:
  01BD8 B7              OR      A               ;End of line reached ?
  01BD9 CA7D1C          JP      Z,1C7DH         ;Yes: continue at 1C7DH
                                                ;No:
  01BDC 3AB040          LD      A,(40B0H)       ;Tokenizing allowed ?
  01BDF B7              OR      A
  01BE0 7E              LD      A,(HL)          ;A = text character
  01BE1 C25B1C          JP      NZ,1C5BH        ;No: accept character
                                                ;Yes:
  01BE4 FE3F            CP      '?'             ;Is it '?'
  01BE6 3EB2            LD      A,0B2H          ;A = PRINT token
  01BE8 CA5B1C          JP      Z,1C5BH         ;Yes: store PRINT token
                                                ;No:
  01BEB 7E              LD      A,(HL)          ;A = text character
  01BEC FE30            CP      '0'             ;Is it a digit ?
  01BEE 3805            JR      C,1BF5H         ;No: tokenize character
                                                ;Yes:
  01BF0 FE3C            CP      3CH             ;Character before '<' ?
  01BF2 DA5B1C          JP      C,1C5BH         ;Yes: accept character (the
                                                ;characters '<' , '=' and '>'
                                                ;are tokenized)
; Tokenize character / text

  01BF5 D5              PUSH    DE              ;Save buffer pointer
  01BF6 114F16          LD      DE,164FH        ;DE -> Keyword table
  01BF9 C5              PUSH    BC              ;Save counter
  01BFA 013D1C          LD      BC,1C3DH        ;Set new RET address
  01BFD C5              PUSH    BC              ;to 1C3DH
  01BFE 067F            LD      B,7FH           ;B = token counter
  01C00 7E              LD      A,(HL)          ;A = text character
  01C01 FE61            CP      'a'             ;Character < 'a' ?
  01C03 3807            JR      C,1C0CH         ;Yes: continue at 1C0CH

  01C05 FE7B            CP      'z'+1           ;Character > 'z' ?
  01C07 3003            JR      NC,1C0CH        ;Yes: continue at 1C0CH

  01C09 E65F            AND     5FH             ;Convert to upper case
  01C0B 77              LD      (HL),A          ;Store character

  01C0C 4E              LD      C,(HL)          ;C = text character
  01C0D EB              EX      DE,HL           ;HL = table pointer

  01C0E 23              INC     HL              ;Table pointer + 1
  01C0F B6              OR      (HL)            ;Next keyword reached ?
  01C10 F20E1C          JP      P,1C0EH         ;No: increment HL to next
                                                ;keyword
  01C13 04              INC     B               ;Yes: token counter + 1
  01C14 7E              LD      A,(HL)          ;A = table character
```

```
  01C15 CDE238           CALL    38E2H            ;Intercept Colour BASIC
                                                  ;keywords
  01C18 B9               CP      C                ;Compare text character with
                                                  ;table character; the same ?
  01C19 20F3             JR      NZ,1C0EH         ;No: increment table pointer
                                                  ;to next keyword and check
                                                  ;again
  01C1B EB               EX      DE,HL            ;DE = table pointer
                                                  ;HL = buffer pointer on first
                                                  ;character
  01C1C E5               PUSH    HL               ;Save buffer pointer
  01C1D 13               INC     DE               ;Get next character
  01C1E 1A               LD      A,(DE)           ;from table
  01C1F B7               OR      A                ;Next keyword reached ?
  01C20 FA391C           JP      M,1C39H          ;Yes: Put token in B
                                                  ;No:
  01C23 4F               LD      C,A              ;C = table character
  01C24 78               LD      A,B              ;A = token counter
  01C25 FE8D             CP      8DH              ;At GOTO token ?
  01C27 2002             JR      NZ,1C2BH         ;No: continue at 1C2BH

  01C29 D7               RST     10H              ;Increment text pointer to next
                                                  ;character (GOTO can also be
                                                  ;written as GO TO !)
  01C2A 2B               DEC     HL               ;HL - 1 because of HL + 1
                                                  ;in the RST 10H
  01C2B 23               INC     HL               ;Text pointer + 1
  01C2C 7E               LD      A,(HL)           ;A = next character
  01C2D FE61             CP      'a'              ;Lower case ?
  01C2F 3802             JR      C,1C33H          ;No: ok, continue at 1C33H

  01C31 E65F             AND     5FH              ;Convert to upper case

  01C33 B9               CP      C                ;Text character the same as
                                                  ;table character
  01C34 28E7             JR      Z,1C1DH          ;Yes: compare next character
                                                  ;No:
  01C36 E1               POP     HL               ;Set text pointer back to first
                                                  ;character
  01C37 18D3             JR      1C0CH            ;And compare with next keyword

; Compare ended succesfully

  01C39 48               LD      C,B              ;C = token
  01C3A F1               POP     AF               ;Remove text pointer from stack
  01C3B EB               EX      DE,HL            ;DE = text pointer
  01C3C C9               RET

; Store token or character

  01C3D EB               EX      DE,HL            ;HL = text pointer
  01C3E 79               LD      A,C              ;A = token
  01C3F C1               POP     BC               ;Restore character counter
  01C40 D1               POP     DE               ;Restore buffer pointer
  01C41 EB               EX      DE,HL            ;DE = text pointer
                                                  ;HL = buffer pointer
  01C42 FE95             CP      95H              ;'ELSE' token ?
  01C44 363A             LD      (HL),3AH         ;Put ':' in buffer
  01C46 2002             JR      NZ,1C4AH         ;No: do not use ':'
                                                  ;Yes:
  01C48 0C               INC     C                ;Counter + 1
  01C49 23               INC     HL               ;Buffer pointer + 1:
                                                  ; ':' is used
  01C4A FEFB             CP      0FBH             ;Apostroph ? (REM)
  01C4C 200C             JR      NZ,1C5AH         ;No: implement token
                                                  ;Yes:
  01C4E 363A             LD      (HL),':'         ;Put ':' in buffer
  01C50 23               INC     HL               ;Buffer poiner + 1
  01C51 0693             LD      B,93H            ;B = 'REM' token
  01C53 70               LD      (HL),B           ;Put token in buffer
  01C54 23               INC     HL               ;Buffer pointer + 1
```

```
  01C55 EB             EX     DE,HL          ;DE = buffer pointer
  01C56 0C             INC    C              ;Counter + 1 (for ':')
  01C57 0C             INC    C              ;Counter + 1 (for 'REM')
  01C58 181D           JR     1C77H          ;Use all characters until end
                                             ;of line (REM line)
; Use token or character in A

  01C5A EB             EX     DE,HL          ;DE = buffer pointer
  01C5B 23             INC    HL             ;Text pointer + 1
  01C5C 12             LD     (DE),A         ;Store token in buffer
  01C5D 13             INC    DE             ;Buffer pointer + 1
  01C5E 0C             INC    C              ;Counter + 1
  01C5F D63A           SUB    ':'            ;':' ? (SUBtract !!)
  01C61 2804           JR     Z,1C67H        ;Yes: release tokenizing and
                                             ;tokenize next character
  01C63 FE4E           CP     4EH            ;'DATA' token ?
                                             ;(4EH + 3AH = 88H)
  01C65 2003           JR     NZ,1C6AH       ;No: leave flag, continue at
                                             ;1C6AH
  01C67 32B040         LD     (40B0H),A      ;Yes: block tokenizing (A=4EH)
  01C6A D659           SUB    59H            ;'REM' token ? (SUBtract !!)
  01C6C C2CC1B         JP     NZ,1BCCH       ;No: tokenize next character
                                             ;Yes:
  01C6F 47             LD     B,A            ;Set B = 0 as compare char.

; Use all characters until end of line or until character = B

  01C70 7E             LD     A,(HL)         ;A = text character
  01C71 B7             OR     A              ;End of line reached ?
  01C72 2809           JR     Z,1C7DH        ;Yes: done, continue ar 1C7DH
                                             ;No:
  01C74 B8             CP     B              ;Same as compare character ?
  01C75 28E4           JR     Z,1C5BH        ;Yes: implement character.
                                             ;No:
  01C77 23             INC    HL             ;Text pointer + 1
  01C78 12             LD     (DE),A         ;Store character in buffer
  01C79 0C             INC    C              ;Counter + 1
  01C7A 13             INC    DE             ;Buffer pointer + 1
  01C7B 18F3           JR     1C70H          ;Next character

; Line completely tokenized

  01C7D 210500         LD     HL,0005H       ;Add 5 bytes to total length
  01C80 44             LD     B,H            ;(2 bytes line pointer, 2 bytes
  01C81 09             ADD    HL,BC          ;line number, 1 byte line end)
  01C82 44             LD     B,H            ;BC = number of bytes required
  01C83 4D             LD     C,L            ;in memory
  01C84 2AA740         LD     HL,(40A7H)     ;HL -> line buffer
  01C87 2B             DEC    HL             ;HL -> tokenized line - 1
  01C88 2B             DEC    HL
  01C89 2B             DEC    HL
  01C8A 12             LD     (DE),A         ;Terminate tokenized line
  01C8B 13             INC    DE             ;with 3 times 00H
  01C8C 12             LD     (DE),A
  01C8D 13             INC    DE
  01C8E 12             LD     (DE),A
  01C8F C9             RET


; SUB RST 18H: 16 bit compare
; Compares DE with HL and set flags according to result
; (like CP HL,DE)

  01C90 7C             LD     A,H            ;A = MSB of HL
  01C91 92             SUB    D              ;Subtract MSB of DE
                                             ;H = D ?
  01C92 C0             RET    NZ             ;No: return

  01C93 7D             LD     A,L            ;Same with LSBs
  01C94 93             SUB    E
  01C95 C9             RET
```

```
; SUB RST 08H: syntax check
; Compare the byte in (HL) with the byte following the RST 08H in memory.
; If they are equal then a RST 10H is executed and the routine returns to
; the normal RET-address + 1 (because the compare byte is located at the
; RET-address). If both bytes are not equal a ?SN Error is generated

  01C96 7E              LD      A,(HL)          ;A = text character
  01C97 E3              EX      (SP),HL         ;HL -> compare character
  01C98 BE              CP      (HL)            ;The same ?
  01C99 23              INC     HL              ;HL + 1 for RET-address
  01C9A E3              EX      (SP),HL         ;RET-address on stack
  01C9B CA781D          JP      Z,1D78H         ;Yes: equal so RST 10H
                                                ;No:
  01C9E C39719          JP      1997H           ;?SN Error


; FOR statement
; -------------

  01CA1 3E64            LD      A,64H           ;A <> 0
  01CA3 32DC40          LD      (40DCH),A       ;Block array variables
  01CA6 CD211F          CALL    1F21H           ;Execute LET (set loop
                                                ;variable and initialize it)
  01CA9 E3              EX      (SP),HL         ;Save PTP, HL = RET-address
                                                ;DE = VARPTR on loop variable
  01CAA CD3619          CALL    1936H           ;Already a FOR-TO-loop active
                                                ;using this variable ?
  01CAD D1              POP     DE              ;Restore PTP
  01CAE 2005            JR      NZ,1CB5H        ;No: continue at 1CB5H
                                                ;Yes:
  01CB0 09              ADD     HL,BC           ;HL = HL + 14 (BC = 000EH)
  01CB1 F9              LD      SP,HL           ;Terminate previous loop using
                                                ;this variable (delete from
                                                ;stack)
  01CB2 22E840          LD      (40E8H),HL      ;Save new SP
  01CB5 EB              EX      DE,HL           ;HL = PTP
  01CB6 0E08            LD      C,08H           ;Are there 16 bytes (2*C) free?
  01CB8 CD6319          CALL    1963H           ;(17 bytes are needed)
  01CBB E5              PUSH    HL              ;Save PTP
  01CBC CD051F          CALL    1F05H           ;Increment PTP to next command
                                                ;(PTP is then pointing on the
                                                ;first command within the loop)
  01CBF E3              EX      (SP),HL         ;Save new PTP, HL = old PTP
  01CC0 E5              PUSH    HL              ;Save PTP
  01CC1 2AA240          LD      HL,(40A2H)      ;HL= current LN
  01CC4 E3              EX      (SP),HL         ;Save LN, restore PTP
  01CC5 CF              RST     08H             ;Next byte must be the
  01CC6 BD              DEFB    0BDH            ;'TO' token
  01CC7 E7              RST     20H             ;TSTTYP
                                                ;STR type ?
  01CC8 CAF60A          JP      Z,0AF6H         ;Yes: ?TM Error
                                                ;DBL type ?
  01CCB D2F60A          JP      NC,0AF6H        ;Yes: ?TM Error

  01CCE F5              PUSH    AF              ;Save type code - 3
  01CCF CD3723          CALL    2337H           ;X = final value of loop
  01CD2 F1              POP     AF              ;Restore type code
  01CD3 E5              PUSH    HL              ;Save PTP
                                                ;Loop in SNG format ?
  01CD4 F2EC1C          JP      P,1CECH         ;Yes: continue at 1CECH

; INT loop

  01CD7 CD7F0A          CALL    0A7FH           ;Convert final value to INT
  01CDA E3              EX      (SP),HL         ;Save final value, restore PTP
  01CDB 110100          LD      DE,0001H        ;DE = default step value (1)
  01CDE 7E              LD      A,(HL)          ;A = next character
  01CDF FECC            CP      0CCH            ;'STEP' token ?
  01CE1 CC012B          CALL    Z,2B01H         ;Yes: DE = step value
```

```
  01CE4 D5              PUSH    DE              ;Save step value
  01CE5 E5              PUSH    HL              ;Save PTP
  01CE6 EB              EX      DE,HL           ;HL = step value
  01CE7 CD9E09          CALL    099EH           ;A = SGN (step value)
  01CEA 1822            JR      1D0EH           ;Continue at 1D0EH

; SNG loop

  01CEC CDB10A          CALL    0AB1H           ;Convert final value to SGN
  01CEF CDBF09          CALL    09BFH           ;BCDE = X = final value
  01CF2 E1              POP     HL              ;Restore PTP
  01CF3 C5              PUSH    BC              ;Save final value
  01CF4 D5              PUSH    DE
  01CF5 010081          LD      BC,8100H        ;BCDE = default step value (1)
  01CF8 51              LD      D,C
  01CF9 5A              LD      E,D
  01CFA 7E              LD      A,(HL)          ;A = next character
  01CFB FECC            CP      0CCH            ;'STEP' token ?
  01CFD 3E01            LD      A,01H           ;A = SNG (default step value)
  01CFF 200E            JR      NZ,1D0FH        ;No: continue at 1D0FH
                                                ;Yes:
  01D01 CD3823          CALL    2338H           ;X = step value
  01D04 E5              PUSH    HL              ;Save PTP
  01D05 CDB10A          CALL    0AB1H           ;X = CSGN (X)
  01D08 CDBF09          CALL    09BFH           ;BCDE = X = step value
  01D0B CD5509          CALL    0955H           ;A = SNG(step value)

; FOR stack wrap up

  01D0E E1              POP     HL              ;Restore PTP
  01D0F C5              PUSH    BC              ;Save step value
  01D10 D5              PUSH    DE              ;(for SNG loop)
  01D11 4F              LD      C,A             ;C = SGN (step value)
  01D12 E7              RST     20H             ;TSTTYP (step value)
  01D13 47              LD      B,A             ;B = type code - 3
  01D14 C5              PUSH    BC              ;Save type code and
                                                ;SNG (step value)
  01D15 E5              PUSH    HL              ;Save PTP
  01D16 2ADF40          LD      HL,(40DFH)      ;HL = VARPTR of loop variable
  01D19 E3              EX      (SP),HL         ;Save VARPTR, restore PTP
  01D1A 0681            LD      B,81H           ;B = 'FOR' token
  01D1C C5              PUSH    BC              ;Mark stack
  01D1D 33              INC     SP              ;Remove LSB

; Program loop
; Return address after execution of a command in the program
; HL (PTP) must point to end of command (':') to end of line (00H)

  01D1E CD5803          CALL    0358H           ;Get key
  01D21 B7              OR      A               ;Key pressed ?
  01D22 C4A01D          CALL    NZ,1DA0H        ;Yes: <SHIFT>+<@> or <BREAK> ?
  01D25 22E640          LD      (40E6H),HL      ;Save PTP
  01D28 ED73E840        LD      (40E8H),SP      ;Save SP
  01D2C 7E              LD      A,(HL)          ;A = next character
  01D2D FE3A            CP      3AH             ;= ':' ?
  01D2F 2829            JR      Z,1D5AH         ;Yes: ok, continue at 1DA5H
                                                ;No:
  01D31 B7              OR      A               ;= 00H ? (end of line)
  01D32 C29719          JP      NZ,1997H        ;No: ?SN Error

; Start new line

  01D35 23              INC     HL              ;PTP + 1

  01D36 7E              LD      A,(HL)          ;Test pointer to next line
  01D37 23              INC     HL
  01D38 B6              OR      (HL)            ;= 0000H (end of program) ?
  01D39 CA7E19          JP      Z,197EH         ;Yes: end program.
                                                ;No:
  01D3C 23              INC     HL              ;DE = line number
  01D3D 5E              LD      E,(HL)
```

```
  01D3E 23              INC     HL
  01D3F 56              LD      D,(HL)
  01D40 EB              EX      DE,HL           ;HL = LN, DE = PTP
  01D41 22A240          LD      (40A2H),HL      ;Store current LN
  01D44 3A1B41          LD      A,(411BH)       ;TRACE active ?
  01D47 B7              OR      A               ;(411BH) <> 0 ?
  01D48 280F            JR      Z,1D59H         ;No: continue at 1D59H

; Execute TRACE

  01D4A D5              PUSH    DE              ;Save PTP
  01D4B 3E3C            LD      A,3CH           ;A = '<'
  01D4D CD2A03          CALL    032AH           ;Print it
  01D50 CDAF0F          CALL    0FAFH           ;Print HL as decimal number
                                                ;(line number)
  01D53 3E3E            LD      A,3EH           ;A = '>'
  01D55 CD2A03          CALL    032AH           ;Print it
  01D58 D1              POP     DE              ;Restore PTP
  01D59 EB              EX      DE,HL           ;HL = PTP, DE = LN

  01D5A D7              RST     10H             ;Next character to A
  01D5B 111E1D          LD      DE,1D1EH        ;Set new RET address
  01D5E D5              PUSH    DE              ;to 1D1EH
  01D5F C8              RET     Z               ;RET when end of line reached

  01D60 D680            SUB     80H             ;Token found ?
  01D62 DA211F          JP      C,1F21H         ;No: interpret character as
                                                ;a variable, continue at LET
  01D65 FE3C            CP      3CH             ;Statement or function ?
  01D67 C3C039          JP      39C0H           ;Colour-keyword found ?

  01D6A 07              RLCA                    ;A = token-number * 2
  01D6B 4F              LD      C,A             ;BC = offset for address table
  01D6C 0600            LD      B,00H
  01D6E EB              EX      DE,HL           ;DE = PTP
  01D6F 212218          LD      HL,1822H        ;HL -> address table
  01D72 09              ADD     HL,BC           ;Add offset
  01D73 4E              LD      C,(HL)          ;Load command address in BC
  01D74 23              INC     HL
  01D75 46              LD      B,(HL)
  01D76 C5              PUSH    BC              ;Put address on stack for RET
  01D77 EB              EX      DE,HL           ;HL = PTP


; SUB RST 10H: increment PTP to next character <> 20H (space)
;
; I: HL = PTP
; O: HL = PTP (+1 at least)
;    A = character at (HL)
;    C-flag = 1 if digit found
;    Z-flag = 1 if end of command (':') or end of line (00H) found

  01D78 23              INC     HL              ;PTP + 1
  01D79 7E              LD      A,(HL)          ;A = next character
  01D7A FE3A            CP      ':'             ;Character > digit ?
  01D7C D0              RET     NC              ;Yes: done (Z-flag = 1 in
                                                ;case of ':')
  01D7D FE20            CP      20H             ;Space ?
  01D7F CA781D          JP      Z,1D78H         ;Yes: get next character

  01D82 FE0B            CP      0BH             ;Character > 0AH ?
  01D84 3005            JR      NC,1D8BH        ;Yes: continue at 1D8BH

  01D86 FE09            CP      09H             ;Character > 08H ?
  01D88 D2781D          JP      NC,1D78H        ;Yes: get next character

  01D8B FE30            CP      '0'             ;Digit found?
  01D8D 3F              CCF                     ;Yes: C-flag = 1
  01D8E 3C              INC     A               ;End of line reached ?
  01D8F 3D              DEC     A               ;(A = 00H)
                                                ;Yes: Z-flag = 1
```

```
  01D90 C9              RET

; RESTORE statement
; -----------------

  01D91 EB              EX      DE,HL           ;DE = PTP
  01D92 2AA440          LD      HL,(40A4H)      ;HL = start of program
  01D95 2B              DEC     HL              ;Hl = HL - 1
  01D96 22FF40          LD      (40FFH),HL      ;DATA pointer =
                                                ;start of program - 1
  01D99 EB              EX      DE,HL           ;HL = PTP
  01D9A C9              RET

; <SHIFT>+<@> or <BREAK> pressed ?

  01D9B CD5803          CALL    0358H           ;Get key
  01D9E B7              OR      A               ;Key pressed ?
  01D9F C8              RET     Z               ;No: return

  01DA0 FE60            CP      60H             ;<SHIFT>+<@> pressed ?
  01DA2 CC8403          CALL    Z,0384H         ;Yes: freeze until new key
                                                ;pressed.
  01DA5 329940          LD      (4099H),A       ;Save ASCII code of last key
  01DA8 3D              DEC     A               ;<BREAK> pressed ? (01H)

; STOP statement
; --------------

  01DA9 C0              RET     NZ              ;No: done, return or ?SN Error

; <BREAK> pressed

  01DAA 3C              INC     A               ;A = 01H (key code of BREAK)
  01DAB C3B41D          JP      1DB4H           ;Continue at 1DB4H

; END statement
; -------------

  01DAE C0              RET     NZ              ;?SN Error ?

  01DAF F5              PUSH    AF              ;Save A (A = 00H !)
  01DB0 CCBB41          CALL    Z,41BBH         ;DOS
  01DB3 F1              POP     AF              ;Restore A

  01DB4 22E640          LD      (40E6H),HL      ;Store PTP in system RAM
  01DB7 21B540          LD      HL,40B5H        ;Reset string table pointer
  01DBA 22B340          LD      (40B3H),HL
  01DBD 21F6FF          LD      HL,0FFF6H       ;--

; Entry for STOP in case <BREAK> was pressed during INPUT

* 01DBE   F6FF          OR      0FFH            ;A <> 0, Z-flag = 0
  01DC0 C1              POP     BC              ;Remove RET-address
  01DC1 2AA240          LD      HL,(40A2H)      ;HL = current LN
  01DC4 E5              PUSH    HL              ;Save current LN
  01DC5 F5              PUSH    AF              ;Save flags
  01DC6 7D              LD      A,L             ;LN = 65535 ?
  01DC7 A4              AND     H               ;(Active command mode)
  01DC8 3C              INC     A
  01DC9 2809            JR      Z,1DD4H         ;Yes: no CONT possible
                                                ;No:
  01DCB 22F540          LD      (40F5H),HL      ;Store LN and PTP for CONT
  01DCE 2AE640          LD      HL,(40E6H)
  01DD1 22F740          LD      (40F7H),HL
  01DD4 CD8B03          CALL    038BH           ;End output to printer
  01DD7 CDF920          CALL    20F9H           ;Start new line
  01DDA F1              POP     AF              ;Restore flags
```

```
  01DDB 213019          LD      HL,1930H        ;HL -> 'Break'
                                                ;STOP ?
  01DDE C2061A          JP      NZ,1A06H        ;Yes: continue at 1A06H
                                                ;No:
  01DE1 C3181A          JP      1A18H           ;END, back to active command
                                                ;mode

; CONT statement
; --------------

  01DE4 2AF740          LD      HL,(40F7H)      ;HL = previous PTP
  01DE7 7C              LD      A,H             ;HL = 0000H ?
  01DE8 B5              OR      L
  01DE9 1E20            LD      E,20H           ;E = error code for ?CN Error
  01DEB CAA219          JP      Z,19A2H         ;Yes: ?CN Error

  01DEE EB              EX      DE,HL           ;DE = PTP
  01DEF 2AF540          LD      HL,(40F5H)      ;HL = previous LN
  01DF2 CDA038          CALL    38A0H           ;Save as current LN and program
                                                ;CRTC on last values
  01DF5 EB              EX      DE,HL           ;HL = PTP
  01DF6 C9              RET                     ;Execute next command


; TRON statement
; --------------

  01DF7 3EAF            LD      A,AFH           ;A <> 0  (trace on)


; TROFF statement
; ---------------

* 01DF8   AF            XOR     A               ;A = 0  (trace off)
  01DF9 321B41          LD      (411BH),A       ;Update TRON/TROFF flag
  01DFC C9              RET


  01DFD F1              POP     AF              ;--
  01DFE E1              POP     HL
  01DFF C9              RET


; DEFSTR statement
; ----------------

  01E00 1E03            LD      E,03H           ;E = VT for STR
  01E02 011E02          LD      BC,021EH        ;--


; DEFINT statement
; ----------------

* 01E03   1E02          LD      E,02H           ;E = VT for INT
  01E05 011E04          LD      BC,041EH        ;--


; DEFSNG statement
; ----------------

* 01E06   1E04          LD      E,04H           ;E = VT for SNG
  01E08 011E08          LD      BC,081EH        ;--


; DEFDBL statement
; ----------------

* 01E09   1E08          LD      E,08H           ;E = VT for DBL
  01E0B CD3D1E          CALL    1E3DH           ;Check if valid var letter
  01E0E 019719          LD      BC,1997H        ;First set new return address
  01E11 C5              PUSH    BC              ;to 1997H (= ?SN Error)
```

```
  01E12 D8                  RET     C                ;Invalid character: return

  01E13 D641                SUB     'A'              ;A = offset for table
  01E15 4F                  LD      C,A              ;C = A
  01E16 47                  LD      B,A              ;B = A
  01E17 D7                  RST     10H              ;Get next character
  01E18 FECE                CP      0CEH             ;Token for '-' ?
  01E1A 2009                JR      NZ,1E25H         ;No: set table
                                                     ;Yes:
  01E1C D7                  RST     10H              ;Get 2nd letter
  01E1D CD3D1E              CALL    1E3DH            ;Letter found ?
  01E20 D8                  RET     C                ;No: ?SN Error
                                                     ;Yes
  01E21 D641                SUB     'A'              ;A = offset for table
  01E23 47                  LD      B,A              ;B = A
  01E24 D7                  RST     10H              ;Increment PTP to next char.

  01E25 78                  LD      A,B              ;A = offset of 2nd letter
  01E26 91                  SUB     C                ;- offset of 1st letter
  01E27 D8                  RET     C                ;?SN Error if the letters were
                                                     ;not in alphabetical order
  01E28 3C                  INC     A                ;A = counter
  01E29 E3                  EX      (SP),HL          ;Save PTP, remove RET-address
  01E2A 210141              LD      HL,4101H         ;HL -> table
  01E2D 0600                LD      B,00H            ;BC = offset to 1st letter
  01E2F 09                  ADD     HL,BC            ;HL -> 1st letter entry
  01E30 73                  LD      (HL),E           ;Set VT in table
  01E31 23                  INC     HL               ;Pointer + 1
  01E32 3D                  DEC     A                ;Counter - 1
  01E33 20FB                JR      NZ,1E30H         ;Set next table entry

  01E35 E1                  POP     HL               ;Restore PTP
  01E36 7E                  LD      A,(HL)           ;A = next character
  01E37 FE2C                CP      ','              ;Comma indicated ?
  01E39 C0                  RET     NZ               ;No: done, return
                                                     ;Yes:
  01E3A D7                  RST     10H              ;Get next character
  01E3B 18CE                JR      1E0BH            ;And repeat DEF


; Test if ASCII character at (HL) is an upper case letter (A-Z)
;
; I: HL -> ASCII character
; O: A = ASCII value of character
;    C-flag is 1 if not A-Z

  01E3D 7E                  LD      A,(HL)           ;A = character
  01E3E FE41                CP      'A'              ;Can it be an letter ?
  01E40 D8                  RET     C                ;No: return

  01E41 FE5B                CP      'Z'+1            ;Is it a upper case letter ?
  01E43 3F                  CCF                      ;No: C-flag = 1
  01E44 C9                  RET


; Convert argument at (HL) to INT number. If negative number then ?FC Error

  01E45 D7                  RST     10H              ;Increment PTP
  01E46 CD022B              CALL    2B02H            ;Get argument
  01E49 F0                  RET     P                ;If positive then return

; ?FC Error

  01E4A 1E08                LD      E,08H            ;E = error code for ?FC Error
  01E4C C3A219              JP      19A2H            ;Continue at error routine


; Decode line number (number of '.')

  01E4F 7E                  LD      A,(HL)           ;A = character
  01E50 FE2E                CP      '.'              ;'.' found ?
```

```
  01E52 EB              EX      DE,HL
  01E53 2AEC40          LD      HL,(40ECH)      ;DE = '.'-LN
  01E56 EB              EX      DE,HL
  01E57 CA781D          JP      Z,1D78H         ;Yes: execute RST 10H, done

; Decode number at (HL) into DE

  01E5A 2B              DEC     HL              ;PTP - 1 (because of RST 10H)
  01E5B 110000          LD      DE,0000H        ;Result = 0
  01E5E D7              RST     10H             ;Get next character
                                                ;Digit found ?
  01E5F D0              RET     NC              ;No: done, return

  01E60 E5              PUSH    HL              ;Save PTP
  01E61 F5              PUSH    AF              ;Save digit
  01E62 219819          LD      HL,1998H        ;HL = 6552
  01E65 DF              RST     18H             ;Result already larger than
                                                ;6552 ?
  01E66 DA9719          JP      C,1997H         ;Yes: ?SN Error (with the
                                                ;current digit DE would become
                                                ;larger than 65529)
  01E69 62              LD      H,D             ;HL = previous result
  01E6A 6B              LD      L,E
  01E6B 19              ADD     HL,DE           ;HL = HL * 10
  01E6C 29              ADD     HL,HL
  01E6D 19              ADD     HL,DE
  01E6E 29              ADD     HL,HL
  01E6F F1              POP     AF              ;Restore digit
  01E70 D630            SUB     '0'             ;Convert to numberical value
  01E72 5F              LD      E,A             ;DE = value
  01E73 1600            LD      D,00H
  01E75 19              ADD     HL,DE           ;Add value
  01E76 EB              EX      DE,HL           ;DE = new result
  01E77 E1              POP     HL              ;Restore PTP
  01E78 18E4            JR      1E5EH           ;Get next digit


; CLEAR statement
; ---------------
                                                ;Argument indicated ?
  01E7A CA611B          JP      Z,1B61H         ;No: continue ar 1B61H

; CLEAR with argument

  01E7D CD461E          CALL    1E46H           ;Get INT-argument,
                                                ;if argument < 0 then ?FC Error
  01E80 2B              DEC     HL              ;PTP - 1
  01E81 D7              RST     10H             ;End of command reached ?
  01E82 C0              RET     NZ              ;No: ?SN Error

  01E83 E5              PUSH    HL              ;Save PTP
  01E84 2AB140          LD      HL,(40B1H)      ;HL = TOPMEM
  01E87 7D              LD      A,L             ;DE -> TOPMEM - (size of new
  01E88 93              SUB     E               ;string space)
  01E89 5F              LD      E,A
  01E8A 7C              LD      A,H
  01E8B 9A              SBC     A,D
  01E8C 57              LD      D,A
                                                ;String space too large ?
  01E8D DA7A19          JP      C,197AH         ;Yes: ?OM Error

  01E90 2AF940          LD      HL,(40F9H)      ;HL -> program end
  01E93 012800          LD      BC,0028H        ;BC = 40
  01E96 09              ADD     HL,BC           ;HL -> program end + 40
  01E97 DF              RST     18H             ;Compare HL and DE
                                                ;At least 40 bytes remain free?
  01E98 D27A19          JP      NC,197AH        ;No: ?OM error

  01E9B EB              EX      DE,HL           ;HL -> new start of string
                                                ;space
  01E9C 22A040          LD      (40A0H),HL      ;Save it in system RAM
```

```
  01E9F E1              POP     HL              ;Restore PTP
  01EA0 C3611B          JP      1B61H           ;Execute CLEAR, done.


; RUN statement
; -------------
                                                ;Line number indicated ?
  01EA3 CA5D1B          JP      Z,1B5DH         ;No: continue at 1B5DH
  01EA6 CDC741          CALL    41C7H           ;DOS
  01EA9 CD611B          CALL    1B61H           ;CLEAR
  01EAC 011E1D          LD      BC,1D1EH        ;BC = RET-address on program
                                                ;loop
  01EAF 1810            JR      1EC1H           ;Continue at GOTO


; GOSUB statement
; ---------------

  01EB1 0E03            LD      C,03H           ;Still 2 * C bytes free ?
  01EB3 CD6319          CALL    1963H           ;If not: ?OM Error
  01EB6 C1              POP     BC              ;BC = RET-address
  01EB7 E5              PUSH    HL              ;Save PTP (for GOSUB stack)
  01EB8 E5              PUSH    HL              ;Save PTP
  01EB9 2AA240          LD      HL,(40A2H)      ;HL = current LN
  01EBC E3              EX      (SP),HL         ;Save current LN, restore PTP
  01EBD 3E91            LD      A,91H           ;A = 'GOSUB' token
  01EBF F5              PUSH    AF              ;Save as marker on stack
  01EC0 33              INC     SP              ;Remove LSB from stack
  01EC1 C5              PUSH    BC              ;Put RET-address back


; GOTO statement
; --------------

  01EC2 CD5A1E          CALL    1E5AH           ;Decode LN, DE = LN
  01EC5 CD071F          CALL    1F07H           ;Increment PTP to end of line
  01EC8 E5              PUSH    HL              ;Save new PTP
  01EC9 2AA240          LD      HL,(40A2H)      ;HL = current LN
  01ECC DF              RST     18H             ;Compare current LN with
                                                ;required LN
  01ECD E1              POP     HL              ;Restore new PTP
  01ECE 23              INC     HL              ;HL = pointer on next line
                                                ;Required LN > current LN ?
  01ECF DC2F1B          CALL    C,1B2FH         ;Yes: search line DE from line
                                                ;HL onwards
  01ED2 D42C1B          CALL    NC,1B2CH        ;No: search line DE from start
                                                ;of program onwards
  01ED5 60              LD      H,B             ;HL = pointer on required line
  01ED6 69              LD      L,C
  01ED7 2B              DEC     HL              ;HL - 1
                                                ;Line found ?
  01ED8 D8              RET     C               ;Yes: done, return

; ?UL Error

  01ED9 1E0E            LD      E,0EH           ;E = error code for ?UL Error
  01EDB C3A219          JP      19A2H           ;Continue at error routine


; RETURN statement
; ----------------

  01EDE C0              RET     NZ              ;?SN Error ?

  01EDF 16FF            LD      D,0FFH          ;Flag <> 0
  01EE1 CD3619          CALL    1936H           ;Search GOSUB stack
  01EE4 F9              LD      SP,HL           ;SP = HL (delete GOSUB stack)
  01EE5 22E840          LD      (40E8H),HL      ;Save new SP in system RAM
  01EE8 FE91            CP      91H             ;GOSUB stack found ?
  01EEA 1E04            LD      E,04H           ;E = error code for ?RG Error
  01EEC C2A219          JP      NZ,19A2H        ;No: ?RG Error
```

```
  01EEF E1                POP     HL               ;HL = LN of gosub line
  01EF0 22A240            LD      (40A2H),HL       ;Save as current LN
  01EF3 23                INC     HL               ;LN = 65535 ?
  01EF4 7C                LD      A,H              ;(GOSUB came from active
  01EF5 B5                OR      L                ;command mode ?)
  01EF6 2007              JR      NZ,1EFFH         ;No: increment PTP to next
                                                   ;command following GOSUB and
                                                   ;continue program execution
                                                   ;from there
  01EF8 3ADD40            LD      A,(40DDH)        ;STOP flag = 0 ?
  01EFB B7                OR      A
  01EFC C2181A            JP      NZ,1A18H         ;No: back to active command
                                                   ;mode
  01EFF 211E1D            LD      HL,1D1EH         ;HL = RET_address for program
                                                   ;loop
  01F02 E3                EX      (SP),HL          ;HL = PTP, RET-address = 1D1EH
  01F03 3EE1              LD      A,0E1H           ;--
* 01F04   E1              POP     HL               ;Restore PTP


; DATA statement
; --------------

; Increment PTP (HL) to next command

  01F05 013A0E            LD      BC,0E3AH         ;C = 3AH (':')


; REM statement
; -------------

; Increment PTP (HL) to next line

* 01F07     0E00          LD      C,00H            ;C = 00H
* 01F08       00          NOP                      ;--
  01F09 0600              LD      B,00H            ;B = 00H
  01F0B 79                LD      A,C              ;A = search character (= 00H
                                                   ;when string found, PTP is then
                                                   ;incremented to end of line
  01F0C 48                LD      C,B              ;C = 00H (search character when
                                                   ;string found)
  01F0D 47                LD      B,A              ;B = search character
  01F0E 7E                LD      A,(HL)           ;A = next character
  01F0F B7                OR      A                ;End of line ?
  01F10 C8                RET     Z                ;Yes: done

  01F11 B8                CP      B                ;Character found ?
  01F12 C8                RET     Z                ;Yes: done

  01F13 23                INC     HL               ;PTP + 1
  01F14 FE22              CP      22H              ;String found ?
  01F16 28F3              JR      Z,1F0BH          ;Yes: increment PTP to end
                                                   ;of line
  01F18 D68F              SUB     8FH              ;'IF' found ?
  01F1A 20F2              JR      NZ,1F0EH         ;No: continue search

  01F1C B8                CP      B                ;If 3AH was searched C-flag = 1
  01F1D 8A                ADC     A,D              ;A = 00H + D + C-flag
  01F1E 57                LD      D,A              ;D = counter for nested
                                                   ;IF THEN ELSE statements
  01F1F 18ED              JR      1F0EH            ;Continue search


; LET statement
; -------------

  01F21 CD0D26            CALL    260DH            ;DE = VARPTR variable
  01F24 CF                RST     08H              ;Next character must be
  01F25 D5                DEFB    0D5H             ;the token for '='
  01F26 EB                EX      DE,HL
```

```
  01F27 22DF40          LD      (40DFH),HL       ;Store VARPTR
  01F2A EB              EX      DE,HL
  01F2B D5              PUSH    DE               ;Save VARPTR
  01F2C E7              RST     20H              ;TSTTYP
  01F2D F5              PUSH    AF               ;Save type code - 3
  01F2E CD3723          CALL    2337H            ;Expression to X
  01F31 F1              POP     AF               ;Restore type code - 3

  01F32 E3              EX      (SP),HL          ;Save PTP, restore VARPTR
  01F33 C603            ADD     A,03H            ;Adjust type code
  01F35 CD1928          CALL    2819H            ;Convert X into desired type
  01F38 CD030A          CALL    0A03H            ;DE -> LSB (X) for SNG, DBL
                                                 ;and INT, TSTTYP
  01F3B E5              PUSH    HL               ;Save VARPTR
                                                 ;STR type ?
  01F3C 2028            JR      NZ,1F66H         ;No: continue at 1FF6H

; Variable assignment to string variable

  01F3E 2A2141          LD      HL,(4121H)       ;HL -> vector of new string
  01F41 E5              PUSH    HL               ;Save pointer
  01F42 23              INC     HL
  01F43 5E              LD      E,(HL)           ;DE -> new string
  01F44 23              INC     HL
  01F45 56              LD      D,(HL)
  01F46 2AA440          LD      HL,(40A4H)       ;HL -> start of BASIC program
  01F49 DF              RST     18H              ;String address < program start?
                                                 ;(string is in line buffer,
                                                 ;e.g. with INPUT)
  01F4A 300E            JR      NC,1F5AH         ;Yes: continue at 1F5AH

  01F4C 2AA040          LD      HL,(40A0H)       ;HL -> start of string space
  01F4F DF              RST     18H              ;String address < string space?
                                                 ;(String constant in program
                                                 ;text)
  01F50 D1              POP     DE               ;Restore string pointer
  01F51 300F            JR      NC,1F62H         ;Yes: continue at 1F62H

  01F53 2AF940          LD      HL,(40F9H)       ;HL -> start of variable space
  01F56 DF              RST     18H              ;Vector address of new string
                                                 ;variable space ? (new string
                                                 ;is a variable)
  01F57 3009            JR      NC,1F62H         ;No: continue at 1F62H

  01F59 3ED1            LD      A,0D1H           ;--
* 01F5A   D1            POP     DE               ;Restore string pointer
  01F5B CDF529          CALL    29F5H            ;BC -> last string in string
                                                 ;table
  01F5E EB              EX      DE,HL            ;DE -> vector of last string in
                                                 ;string table
                                                 ;HL -> vector of new string
  01F5F CD4328          CALL    2843H            ;Put new string in string space
                                                 ;and string vector in string
                                                 ;table
  01F62 CDF529          CALL    29F5H            ;DE -> vector of new string
  01F65 E3              EX      (SP),HL          ;HL -> vector of variables
                                                 ;Save vector of new string
  01F66 CDD309          CALL    09D3H            ;Copy VT bytes from (DE)
                                                 ;to (HL)
  01F69 D1              POP     DE               ;Restore vector to new string
  01F6A E1              POP     HL               ;Restore PTP
  01F6B C9              RET


; ON statement
; ------------

  01F6C FE9E            CP      9EH              ;Next token = 'ERROR'
  01F6E 2025            JR      NZ,1F95H         ;No: continue at 1F95H
```

```
; ON ERROR GOTO

  01F70 D7              RST     10H             ;Get next non-space character
  01F71 CF              RST     08H             ;Next byte must be token
  01F72 8D              DEFB    8DH             ;for 'GOTO'
  01F73 CD5A1E          CALL    1E5AH           ;Decode line number
  01F76 7A              LD      A,D             ;Line number = 0 ?
  01F77 B3              OR      E
  01F78 2809            JR      Z,1F83H         ;Yes: continue at 1F83H

  01F7A CD2A1B          CALL    1B2AH           ;Save PTP and search line DE
  01F7D 50              LD      D,B             ;DE = address of line
  01F7E 59              LD      E,C
  01F7F E1              POP     HL              ;Restore PTP
                                                ;Line found ?
  01F80 D2D91E          JP      NC,1ED9H        ;No: ?UL Error

  01F83 EB              EX      DE,HL
  01F84 22F040          LD      (40F0H),HL      ;Save line number
  01F87 EB              EX      DE,HL
                                                ;Line number <> 0 ?
  01F88 D8              RET     C               ;Yes: done, return

  01F89 3AF240          LD      A,(40F2H)       ;A = error flag
  01F8C B7              OR      A               ;ON ERROR GOTO active ?
  01F8D C8              RET     Z               ;No: done, return

  01F8E 3A9A40          LD      A,(409AH)       ;A = last error code
  01F91 5F              LD      E,A             ;into E
  01F92 C3AB19          JP      19ABH           ;Process error


; ON GOTO / GOSUB

  01F95 CD1C2B          CALL    2B1CH           ;DE = argument (0 - 255)
  01F98 7E              LD      A,(HL)          ;A = next character
  01F99 47              LD      B,A             ;B = A
  01F9A FE91            CP      91H             ;'GOSUB' token ?
  01F9C 2803            JR      Z,1FA1H         ;Yes: continue at 1F1AH
                                                ;No:
  01F9E CF              RST     08H             ;It must be the
  01F9F 8D              DEFB    8DH             ;'GOTO' token
  01FA0 2B              DEC     HL              ;PTP - 1 (because of RST 08H)
  01FA1 4B              LD      C,E             ;C = argument (counter)
  01FA2 0D              DEC     C               ;Counter - 1
  01FA3 78              LD      A,B             ;A = token
                                                ;Counter = 0 ?
  01FA4 CA601D          JP      Z,1D60H         ;Yes: HL points to required LN
                                                ;so execute command
  01FA7 CD5B1E          CALL    1E5BH           ;By decoding the number at (HL)
                                                ;increment PTP to the next
                                                ;character following the number
  01FAA FE2C            CP      ','             ;Separator must be a comma
  01FAC C0              RET     NZ              ;Otherwise execute next command

  01FAD 18F3            JR      1FA2H           ;Required LN reached ?


; RESUME statement
; ----------------

  01FAF 11F240          LD      DE,40F2H        ;DE -> ON ERROR GOT flag
  01FB2 1A              LD      A,(DE)          ;A = ON ERROR GOTO flag
  01FB3 B7              OR      A               ;ON ERROR GOTO active ?
  01FB4 CAA019          JP      Z,19A0H         ;No: ?RW Error

  01FB7 3C              INC     A               ;A + 1
  01FB8 329A40          LD      (409AH),A       ;Set last error code <> 0
  01FBB 12              LD      (DE),A          ;Set error flag <> 0
  01FBC 7E              LD      A,(HL)          ;A = next program character
  01FBD FE87            CP      87H             ;'NEXT' token ?
```

```
  01FBF 280C          JR      Z,1FCDH        ;Yes: continue at 1FCDH

  01FC1 CD5A1E        CALL    1E5AH          ;Decode line number
  01FC4 C0            RET     NZ             ;?SN Error ?

  01FC5 7A            LD      A,D            ;Line number = 0 ?
  01FC6 B3            OR      E
  01FC7 C2C51E        JP      NZ,1EC5H       ;No: execute GOTO
                                             ;Yes:
  01FCA 3C            INC     A              ;A = 1 (Z-flag = 0)
  01FCB 1802          JR      1FCFH          ;Execute RESUME NEXT

; RESUME NEXT

  01FCD D7            RST     10H            ;Increment PTP
  01FCE C0            RET     NZ             ;?SN Error ?

  01FCF 2AEE40        LD      HL,(40EEH)     ;HL = PTP on next command
  01FD2 EB            EX      DE,HL          ;DE = HL
  01FD3 2AEA40        LD      HL,(40EAH)     ;HL = LN of error line
  01FD6 22A240        LD      (40A2H),HL     ;Store as current LN
  01FD9 EB            EX      DE,HL          ;HL = PTP
  01FDA C0            RET     NZ             ;Done if RESUME 0

  01FDB 7E            LD      A,(HL)         ;A = next character from
                                             ;program text
  01FDC B7            OR      A              ;End of line ?
  01FDD 2004          JR      NZ,1FE3H       ;No: continue at 1FE3H
                                             ;Yes:
  01FDF 23            INC     HL             ;Increment PTP to
  01FE0 23            INC     HL             ;next line
  01FE1 23            INC     HL
  01FE2 23            INC     HL
  01FE3 23            INC     HL
  01FE4 7A            LD      A,D            ;Error line = 65535 ?
  01FE5 A3            AND     E
  01FE6 3C            INC     A
  01FE7 C2051F        JP      NZ,1F05H       ;No: increment PTP to next
                                             ;command and continue program
                                             ;execution
  01FEA 3ADD40        LD      A,(40DDH)      ;A = STOP flag
  01FED 3D            DEC     A              ;A = 1 ?
  01FEE CABE1D        JP      Z,1DBEH        ;Yes: STOP
                                             ;No:
  01FF1 C3051F        JP      1F05H          ;Continue program execution
                                             ;at next command

; ERROR statement
; ---------------

  01FF4 CD1C2B        CALL    2B1CH          ;DE = argument (0 - 255)
  01FF7 C0            RET     NZ             ;?SN Error ?

  01FF8 B7            OR      A              ;ERROR 0 ?
  01FF9 CA4A1E        JP      Z,1E4AH        ;YEs: ?FC Error

  01FFC 3D            DEC     A              ;A = A - 1
  01FFD 87            ADD     A,A            ;A = A * 2
  01FFE 5F            LD      E,A            ;E = (argument - 1 ) * 2
  01FFF FE2D          CP      2DH            ;Errorcode > 44H (out of range)
  02001 3802          JR      C,2005H        ;No: process error code
                                             ;Yes:
  02003 1E26          LD      E,26H          ;E = error code for ?UE Error
  02005 C3A219        JP      19A2H          ;Continue at error routine


; AUTO statement
; --------------

  02008 110A00        LD      DE,000AH       ;DE = default first LN
  0200B D5            PUSH    DE             ;Save first LN
```

```
                                                ;Any numbers indicated ?
  0200C 2817            JR      Z,2025H         ;No: continue at 2025H

  0200E CD4F1E          CALL    1E4FH           ;DE = indicated first LN
  02011 EB              EX      DE,HL           ;HL = LN, DE = PTP
  02012 E3              EX      (SP),HL         ;Save first LN, HL = default
                                                ;first LN. Distance indicated ?
  02013 2811            JR      Z,2026H         ;No: continue at 2026H

  02015 EB              EX      DE,HL           ;PTP back to HL
  02016 CF              RST     08H             ;Next character must be
  02017 2C              DEFB    ','             ;a comma
  02018 EB              EX      DE,HL
  02019 2AE440          LD      HL,(40E4H)      ;DE = previous distance
  0201C EB              EX      DE,HL
                                                ;use previous distance ?
                                                ;(AUTO XX, )
  0201D 2806            JR      Z,2025H         ;Yes: continue at 2025H

  0201F CD5A1E          CALL    1E5AH           ;DE = distance
  02022 C29719          JP      NZ,1997H        ;?SN Error

  02025 EB              EX      DE,HL           ;DE = PTP, HL = distance
  02026 7C              LD      A,H             ;Distance =0 ?
  02027 B5              OR      L
  02028 CA4A1E          JP      Z,1E4AH         ;Yes: ?FC Error

  0202B 22E440          LD      (40E4H),HL      ;Store distance in system RAM
  0202E 32E140          LD      (40E1H),A       ;Set AUTO flag <> 0
  02031 E1              POP     HL              ;Restore first LN
  02032 22E240          LD      (40E2H),HL      ;and store it in system RAM
  02035 C1              POP     BC              ;Delete RET-address
  02036 C3331A          JP      1A33H           ;Input line


; IF statement
; ------------

  02039 CD3723          CALL    2337H           ;X = expression
                                                ;(if condition is false then
                                                ;X = 0 else X = -1)
  0203C 7E              LD      A,(HL)          ;Get next character
  0203D FE2C            CP      ','             ;Is it a comma ?
  0203F CC781D          CALL    Z,1D78H         ;Yes: increment PTP (like
                                                ;with 'THEN')
  02042 FECA            CP      0CAH            ;'THEN' token ?
  02044 CC781D          CALL    Z,1D78H         ;Yes: increment PTP

  02047 2B              DEC     HL              ;PTP - 1
  02048 E5              PUSH    HL              ;Save PTP
  02049 CD9409          CALL    0994H           ;TEST1
  0204C E1              POP     HL              ;Restore PTP
                                                ;X = 0 ? (condition: false)
  0204D 2807            JR      Z,2056H         ;Yes: continue at 2056H

; IF-condition is true

  0204F D7              RST     10H             ;Increment PTP
                                                ;Number indicated ?
  02050 DAC21E          JP      C,1EC2H         ;Yes: continue at GOTO
                                                ;No:
  02053 C35F1D          JP      1D5FH           ;execute command token

; IF-condition is false

  02056 1601            LD      D,01H           ;D = counter for nested
                                                ;IF-THEN-ELSE conditions
  02058 CD051F          CALL    1F05H           ;Increment PTP to next command
  0205B B7              OR      A               ;End of line reached ?
  0205C C8              RET     Z               ;Yes: done, return
```

```
  0205D D7                RST     10H             ;A = next character
  0205E FE95              CP      95H             ;'ELSE' token found ?
  02060 20F6              JR      NZ,2058H        ;No: continue search

  02062 15                DEC     D               ;Nesting level counter - 1
  02063 20F3              JR      NZ,2058H        ;Search outermost 'ELSE'

  02065 18E8              JR      204FH           ;Execute following command


; LPRINT statement
; ----------------

  02067 3E01              LD      A,01H           ;Set output flag to
  02069 329C40            LD      (409CH),A       ;printer output
  0206C C39B20            JP      209BH           ;Continue at PRINT


; PRINT statement
; ---------------

  0206F CDCA41            CALL    41CAH           ;DOS
  02072 FE40              CP      40H             ;PRINT@ ?
  02074 2019              JR      NZ,208FH        ;No: continue at 208FH

  02076 CD012B            CALL    2B01H           ;Get argument
  02079 E5                PUSH    HL              ;Save PTP
  0207A C3D430            JP      30D4H           ;Test argument and restore
  0207D 00                NOP                     ;PTP (returns to 207EH)
  0207E E5                PUSH    HL              ;Save PTP
  0207F 210044            LD      HL,4400H        ;HL -> start of screen mem
  02082 19                ADD     HL,DE           ;Add @-argument
  02083 222040            LD      (4020H),HL      ;Store new cursor position
  02086 CD2A36            CALL    362AH           ;Calculate new POS
  02089 32A640            LD      (40A6H),A       ;and save it in system RAM
  0208C E1                POP     HL              ;Restore PTP
  0208D CF                RST     08H             ;After @ must follow
  0208E 2C                DEFB    ','             :a comma.
  0208F FE23              CP      23H             ;PRINT# ?
  02091 2008              JR      NZ,209BH        ;No, continue at 209BH

  02093 CDA935            CALL    35A9H           ;Write leader and sync
  02096 3E80              LD      A,80H           ;Set output flag to
  02098 329C40            LD      (409CH),A       ;cassette output
  0209B 2B                DEC     HL              ;PTP - 1
  0209C D7                RST     10H             ;A = next non space character
  0209D CCFE20            CALL    Z,20FEH         ;End PRINT if no argument
  020A0 CA6921            JP      Z,2169H         ;and set next output to screen

  020A3 FEBF              CP      0BFH            ;'USING' token ?
  020A5 CABD2C            JP      Z,2CBDH         ;Yes: continue at 2CBDH

  020A8 FEBC              CP      0BCH            ;'TAB(' token ?
  020AA CA3721            JP      Z,2137H         ;Yes: continue at 2137H

  020AD E5                PUSH    HL              ;Save PTP
  020AE FE2C              CP      ','             ;',' found ?
  020B0 CA0821            JP      Z,2108H         ;Yes: continue at 2108H

  020B3 FE3B              CP      ';'             ;';' found ?
  020B5 CA6421            JP      Z,2164H         ;Yes: continue at 2164H

  020B8 C1                POP     BC              ;Correct stack

  020B9 CD3723            CALL    2337H           ;Put argument in X
  020BC E5                PUSH    HL              ;Save PTP
  020BD E7                RST     20H             ;TSTTYP
                                                  ;STR type?
  020BE 2832              JR      Z,20F2H         ;Yes: continue at 202FH

  020C0 CDBD0F            CALL    0FBDH           ;Convert number into string
```

```
  020C3 CD6528          CALL    2865H           ;Take string
  020C6 CDCD41          CALL    41CDH           ;DOS
  020C9 2A2141          LD      HL,(4121H)      ;HL -> string vector
  020CC 3A9C40          LD      A,(409CH)       ;A = output flag
  020CF B7              OR      A               ;Test output flag
                                                ;Cassette output ?
  020D0 FAE920          JP      M,20E9H         ;Yes: continue at 20E9H
                                                ;Screen output?
  020D3 2808            JR      Z,20DDH         ;Yes: continue at 20DDH

; Printer output

  020D5 3A9B40          LD      A,(409BH)       ;A = printer-POS
  020D8 86              ADD     A,(HL)          ;A = printer-POS + string len
  020D9 FE84            CP      84H             ;> 132 ?
  020DB 1809            JR      20E6H           ;Yes: start new line

; Screen output

  020DD 3A9D40          LD      A,(409DH)       ;A = maximum number of
                                                ;characters/line
  020E0 47              LD      B,A             ;B = A
  020E1 3AA640          LD      A,(40A6H)       ;A = screen-POS
  020E4 86              ADD     A,(HL)          ;A = screen-POS + string length
  020E5 B8              CP      B               ;> length of 1 line ?
  020E6 D4FE20          CALL    NC,20FEH        ;Yes: start new line

; Cassette output

  020E9 CDAA28          CALL    28AAH           ;Output string
  020EC 3E20            LD      A,20H           ;A = ' '
  020EE CD2A03          CALL    032AH           ;Print it for separation
  020F1 B7              OR      A               ;set Z-flag= 0: skip next CALL
  020F2 CCAA28          CALL    Z,28AAH         ;Print string when argument
                                                ;was in STR format
  020F5 E1              POP     HL              ;Restore PTP
  020F6 C39B20          JP      209BH           ;Process next argument

; Start a new line

  020F9 3AA640          LD      A,(40A6H)       ;A = screen-POS
  020FC B7              OR      A               ;POS = 0 ? (new line already
                                                ;started)
  020FD C8              RET     Z               ;Yes: done

  020FE 3E0D            LD      A,0DH           ;A = Carriage Return
  02100 CD2A03          CALL    032AH           ;Print it
  02103 CDD041          CALL    41D0H           ;DOS
  02106 AF              XOR     A               ;A = 0
  02107 C9              RET

; ',' with PRINT

  02108 CDD341          CALL    41D3H           ;DOS
  0210B 3A9C40          LD      A,(409CH)       ;A = output flag
  0210E B7              OR      A               ;Test output flag
                                                ;Cassette output ?
  0210F F21921          JP      P,2119H         ;No: continue at 2119H

; Cassette output

  02112 3E2C            LD      A,2CH           ;A = ','
  02114 CD2A03          CALL    032AH           ;Print it
  02117 184B            JR      2164H           ;Process next argument

; PRINT ',' on screen or printer

  02119 2808            JR      Z,2123H         ;continue at 2123H if screen
                                                ;output
; Printer output
```

```
  0211B 3A9B40          LD      A,(409BH)       ;A = printer-POS
  0211E FE70            CP      70H             ;A > 112 ?
  02120 C32B21          JP      212BH           ;Yes: start new line

; Screen output

  02123 3A9E40          LD      A,(409EH)       ;A = highest TAB position
  02126 47              LD      B,A             ;B = A
  02127 3AA640          LD      A,(40A6H)       ;A = screen-POS
  0212A B8              CP      B               ;POS above highest TAB
                                                ;TAB position
  0212B D4FE20          CALL    NC,20FEH        ;Yes: start a new line
  0212E 3034            JR      NC,2164H        ;and process next argument

  02130 D60A            SUB     0AH             ;A = A MOD 10
  02132 30FC            JR      NC,2130H
  02134 2F              CPL

  02135 1823            JR      215AH           ;Print A times a space

; PRINT TAB(

  02137 CD1B2B          CALL    2B1BH           ;DE = TAB argument
  0213A CDB230          CALL    30B2H           ;E = new POS value
  0213D CF              RST     08H             ;Next character must be
  0213E 29              DEFB    ')'             ;a ')'
  0213F 2B              DEC     HL              ;PTP - 1
  02140 E5              PUSH    HL              ;Save PTP
  02141 CDD341          CALL    41D3H           ;DOS
  02144 3A9C40          LD      A,(409CH)       ;A = output flag
  02147 B7              OR      A               ;Test output flag
                                                ;Cassette output ?
  02148 FA4A1E          JP      M,1E4AH         ;Yes: ?FC Error
                                                ;Screen output ?
  0214B CA5321          JP      Z,2153H         ;Yes: continue at 2153H

; Printer output

  0214E 3A9B40          LD      A,(409BH)       ;A = printer-POS
  02151 1803            JR      2156H           ;Continue at 2156H

; Screen output

  02153 3AA640          LD      A,(40A6H)       ;A = screen-POS
  02156 2F              CPL                     ;A = -A
  02157 83              ADD     A,E             ;A = E - A
                                                ;(new POS - old POS)
  02158 300A            JR      NC,2164H        ;Done when required POS
                                                ;already reached
  0215A 3C              INC     A               ;A + 1
  0215B 47              LD      B,A             ;into B as counter
  0215C 3E20            LD      A,20H           ;A = ' '
                                                ;Repeat
  0215E CD2A03          CALL    032AH           ;Print space
  02161 05              DEC     B
  02162 20FA            JR      NZ,215EH        ;Until B = 0

; ';' with PRINT, do not generate at new line

  02164 E1              POP     HL              ;Restore PTP
  02165 D7              RST     10H             ;Skip spaces/LF etc and get
                                                ;next character
  02166 C3A020          JP      20A0H           ;Continue at 20A0H

; Next output back to screen

  02169 3A9C40          LD      A,(409CH)       ;--
  0216C B7              OR      A               ;
  0216D 00              NOP
  0216E 00              NOP
  0216F 00              NOP
```

```
  02170 AF              XOR     A                ;Set output flag to
  02171 329C40          LD      (409CH),A        ;screen output
  02174 CDBE41          CALL    41BEH            ;DOS
  02177 C9              RET

; Text '?REDO'

  02178 3F5245444F      DEFB    '?REDO'
  0217D 0D              DEFB    0DH              ;Carriage return
  0217E 00              DEFB    00H              ;End of string


; Data with INPUT or READ not separated by ','

  0217F 3ADE40          LD      A,(40DEH)        ;A = READ/INPUT flag
  02182 B7              OR      A                ;READ?
  02183 C29119          JP      NZ,1991H         ;Yes: ?SN Error

  02186 CDB130          CALL    30B1H            ;Test E on correct TAB
                                                 ;value (???)
  02189 B7              OR      A                ;A = 0 (???)
  0218A 1E2A            LD      E,2AH            ;E = error code for ?FD Error
  0218C CAA219          JP      Z,19A2H          ;Yes: ?FD Error

  0218F C1              POP     BC               ;No: correct stack
  02190 217821          LD      HL,2178H         ;HL -> '?REDO'
  02193 CDA728          CALL    28A7H            ;Print text
  02196 2AE640          LD      HL,(40E6H)       ;HL = last PTP
  02199 C9              RET                      ;Repeat last command


; INPUT statement
; ---------------

  0219A CD2828          CALL    2828H            ;?ID Error?
  0219D 7E              LD      A,(HL)           ;A = next character
  0219E CDD641          CALL    41D6H            ;DOS
  021A1 D623            SUB     23H              ;Character = '#' ?
  021A3 32A940          LD      (40A9H),A        ;Flag = 0 when INPUT#
  021A6 7E              LD      A,(HL)           ;A = character
  021A7 2020            JR      NZ,21C9H         ;No: continue at 21C9H

; INPUT#

  021A9 CDAF35          CALL    35AFH            ;Evaluate number and
                                                 ;search for sync and leader
  021AC E5              PUSH    HL               ;Save PTP
  021AD 06FA            LD      B,0FAH           ;B = 250 (record length)
  021AF 2AA740          LD      HL,(40A7H)       ;HL -> line buffer
  021B2 CDED01          CALL    01EDH            ;Read one byte
  021B5 77              LD      (HL),A           ;Store it in buffer
  021B6 23              INC     HL               ;Pointer + 1
  021B7 FE0D            CP      0DH              ;End of record ?
  021B9 2802            JR      Z,21BDH          ;Yes: continue at 21BDH

  021BB 10F5            DJNZ    21B2H            ;Read next byte

  021BD 2B              DEC     HL               ;Pointer - 1
  021BE 3600            LD      (HL),00H         ;Replace last byte with 00H
  021C0 00              NOP                      ;--
  021C1 00              NOP                      ;(I guess motor off call used
  021C2 00              NOP                      ;to be here in the TRS80)
  021C3 2AA740          LD      HL,(40A7H)       ;HL -> line buffer
  021C6 2B              DEC     HL               ;Pointer - 1
  021C7 1822            JR      21EBH            ;Process data

; Normal INPUT

  021C9 01DB21          LD      BC,21DBH         ;Set new RET address
  021CC C5              PUSH    BC               ;to 21DBH
  021CD FE22            CP      22H              ;Text to be printed first?
```

```
  021CF C0              RET     NZ              ;No: continue at 21DBH

  021D0 CD6628          CALL    2866H           ;Get text
  021D3 CF              RST     08H             ;Text must be followed by
  021D4 3B              DEFB    ';'             ;a ';'
  021D5 E5              PUSH    HL              ;Save PTP
  021D6 CDAA28          CALL    28AAH           ;Print text
  021D9 E1              POP     HL              ;Restore PTP
  021DA C9              RET                     ;Continue at 21BDH

  021DB E5              PUSH    HL              ;Save PTP
  021DC CDB31B          CALL    1BB3H           ;Print '?' and goto line input
  021DF C1              POP     BC              ;BC = PTP
                                                ;<BREAK> pressed ?
  021E0 DABE1D          JP      C,1DBEH         ;Yes: continue at STOP

  021E3 23              INC     HL              ;Pointer + 1
  021E4 7E              LD      A,(HL)          ;Get first character
  021E5 B7              OR      A               ;<RETURN> pressed ?
  021E6 2B              DEC     HL              ;Pointer - 1
  021E7 C5              PUSH    BC              ;Save PTP
  021E8 CA041F          JP      Z,1F04H         ;Yes: set PTP to next
                                                ;instruction and leave
                                                ;variables unchanged
  021EB 362C            LD      (HL),2CH        ;Put a comma in the buffer
                                                ;as a separator
  021ED 1805            JR      21F4H           ;Continue at 21F4H


; READ statement
; --------------

  021EF E5              PUSH    HL              ;Save PTP
  021F0 2AFF40          LD      HL,(40FFH)      ;HL -> next data value
  021F3 F6AF            OR      AFH             ;set flag to READ

* 021F4   AF            XOR     A               ;set flag to INPUT
  021F5 32DE40          LD      (40DEH),A       ;Store READ/INPUT flag
  021F8 E3              EX      (SP),HL         ;Restore PTP, save DATA pointer
  021F9 1802            JR      21FDH           ;Continue at 21FDH


; Use next data
; DATA pointer = pointer on the data to use
; PTP = pointer on variables

  021FB CF              RST     08H             ;Next character must be
  021FC 2C              DEFB    ','             ;a comma (separator)

  021FD CD0D26          CALL    260DH           ;Get address of variable
                                                ;indicated after READ/INPUT
  02200 E3              EX      (SP),HL         ;Save PTP, restore READ pointer
  02201 D5              PUSH    DE              ;Save variable address
  02202 7E              LD      A,(HL)          ;A = data character
  02203 FE2C            CP      ','             ;Separator ?
  02205 2826            JR      Z,222DH         ;Yes: continue at 222DH

; No separator (',') found

  02207 3ADE40          LD      A,(40DEH)       ;A = READ/INPUT flag
  0220A B7              OR      A               ;READ ?
  0220B C29622          JP      NZ,2296H        ;Yes: increment DATA pointer
                                                ;to next DATA line
  0220E 3AA940          LD      A,(40A9H)       ;INPUT# ?
  02211 B7              OR      A
  02212 1E06            LD      E,06H           ;E = error code for ?OD Error
  02214 CAA219          JP      Z,19A2H         ;Yes: issue error

  02217 3E3F            LD      A,3FH           ;A = '?'
  02219 CD2A03          CALL    032AH           ;Print it
  0221C CDB31B          CALL    1BB3H           ;Request missing data ('??')
```

```
  0221F D1              POP     DE              ;Restore variable address
  02220 C1              POP     BC              ;Restore PTP
                                                ;<BREAK> pressed ?
  02221 DABE1D          JP      C,1DBEH         ;Yes: continue at STOP

  02224 23              INC     HL              ;Pointer + 1
  02225 7E              LD      A,(HL)          ;A = first character
  02226 B7              OR      A               ;<RETURN> pressed ?
  02227 2B              DEC     HL              ;Pointer - 1
  02228 C5              PUSH    BC              ;Save PTP
  02229 CA041F          JP      Z,1F04H         ;Yes: set PTP to next
                                                ;instruction and leave
                                                ;variables unchanged
  0222C D5              PUSH    DE              ;No: Save variable address
  0222D CDDC41          CALL    41DCH           ;DOS
  02230 E7              RST     20H             ;TSTTYP
  02231 F5              PUSH    AF              ;Save VT - 3
                                                ;STR type ?
  02232 2019            JR      NZ,224DH        ;No: continue at 224DH

; Put data in string variable

  02234 D7              RST     10H             ;First non-space/LF character
                                                ;to A
  02235 57              LD      D,A             ;D = A
  02236 47              LD      B,A             ;B = A
  02237 FE22            CP      22H             ;String delimited by '"'
  02239 2805            JR      Z,2240H         ;Yes: B = D = string end

  0223B 163A            LD      D,3AH           ;String end can be ':' (3AH)
  0223D 062C            LD      B,2CH           ;or ',' (2CH)
  0223F 2B              DEC     HL              ;Pointer - 1
  02240 CD6928          CALL    2869H           ;Get string
  02243 F1              POP     AF              ;Restore VT - 3
  02244 EB              EX      DE,HL           ;DE = DATA pointer
  02245 215A22          LD      HL,225AH        ;Set RET address to 225AH
  02248 E3              EX      (SP),HL         ;HL -> variable
  02249 D5              PUSH    DE              ;Save DATA pointer
  0224A C3331F          JP      1F33H           ;Copy new value into variable
                                                ;and continue at 225AH
; Put data in numerical variable

  0224D D7              RST     10H             ;Set pointer on first non-space
                                                ;character
  0224E F1              POP     AF              ;Restore VT - 3
  0224F F5              PUSH    AF              ;And save it again
  02250 014322          LD      BC,2243H        ;Set new RET address to 2243H
  02253 C5              PUSH    BC              ;(copy value into variable)
                                                ;INT or SNG value ?
  02254 DA6C0E          JP      C,0E6CH         ;Yes: get INT or SNG value
                                                ;DBL value
  02257 D2650E          JP      NC,0E65H        ;Yes: get DBL VALUE

; Continuation after getting data

  0225A 2B              DEC     HL              ;PTP - 1
  0225B D7              RST     10H             ;End of line reached ?
  0225C 2805            JR      Z,2263H         ;Yes: continue at 2263H

  0225E FE2C            CP      2CH             ;Separator found ?
  02260 C27F21          JP      NZ,217FH        ;No: process error at 217FH

  02263 E3              EX      (SP),HL         ;Save DATA pointer, HL = PTP
  02264 2B              DEC     HL              ;PTP - 1
  02265 D7              RST     10H             ;End of READ/INPUT ?
  02266 C2FB21          JP      NZ,21FBH        ;No: process next variable

  02269 D1              POP     DE              ;Yes: restore DATA pointer
  0226A 00              NOP                     ;--
  0226B 00              NOP
  0226C 00              NOP
```

```
  0226D 00              NOP
  0226E 00              NOP
  0226F 3ADE40          LD      A,(40DEH)       ;A = READ/INPUT flag
  02272 B7              OR      A               ;READ ?
  02273 EB              EX      DE,HL           ;DE = PTP, HL = DATA pointer
  02274 C2961D          JP      NZ,1D96H        ;Yes: continue at 1D96H

  02277 D5              PUSH    DE              ;No: save PTP
  02278 CDDF41          CALL    41DFH           ;DOS
  0227B B6              OR      (HL)            ;Still data remaining in
                                                ;the buffer ?
  0227C 218622          LD      HL,2286H        ;HL -> '?Extra ignored'
  0227F C4A728          CALL    NZ,28A7H        ;Yes: print text
  02282 E1              POP     HL              ;Restore PTP
  02283 C36921          JP      2169H           ;Direct next output to screen


; Text '?Extra ignored'

  02286 3F4578747261    DEFB    '?Extra'
  0228C 20              DEFB    ' '
  0228D 69676E6F726564  DEFB    'ignored'
  02294 0D              DEFB    0DH             ;Carriage Return
  02295 00              DEFB    00H             ;End of string


; Increment DATA pointer to next DATA line

  02296 CD051F          CALL    1F05H           ;Increment HL to end of
                                                ;command or end of line
  02299 B7              OR      A               ;End of line reached ?
  0229A 2012            JR      NZ,22AEH        ;No: continue at 22AEH

  0229C 23              INC     HL              ;Program end reached ?
  0229D 7E              LD      A,(HL)          ;(Line pointer = 0000H ?)
  0229E 23              INC     HL
  0229F B6              OR      (HL)
  022A0 1E06            LD      E,06H           ;E = error code for ?OD Error
  022A2 CAA219          JP      Z,19A2H         ;Yes: issue ?OD Error
                                                ;No:
  022A5 23              INC     HL              ;DE = line number
  022A6 5E              LD      E,(HL)
  022A7 23              INC     HL
  022A8 56              LD      D,(HL)
  022A9 EB              EX      DE,HL           ;Store line number of
  022AA 22DA40          LD      (40DAH),HL      ;DATA line
  022AD EB              EX      DE,HL
  022AE D7              RST     10H             ;DATA line found ?
  022AF FE88            CP      88H             ;First char. = 'DATA' token ?
  022B1 20E3            JR      NZ,2296H        ;No: continue search
                                                ;Yes:
  022B3 C32D22          JP      222DH           ;HL = new DATA pointer


; NEXT statement
; --------------

  022B6 110000          LD      DE,0000H        ;Default address in case no
                                                ;loop variable is indicated
  022B9 C40D26          CALL    NZ,260DH        ;Get variable address in case
                                                ;loop variable is indicated
  022BC 22DF40          LD      (40DFH),HL      ;Save PTP
  022BF CD3619          CALL    1936H           ;Search FOR stack with
                                                ;VARPTR = DE (next FOR stack
                                                ;in case DE = 0)
                                                ;FOR stack found ?
  022C2 C29D19          JP      NZ,199DH        ;No: ?NF Error
                                                ;Yes:
  022C5 F9              LD      SP,HL           ;SP = FOR stack pointer
  022C6 22E840          LD      (40E8H),HL      ;Save SP value
  022C9 D5              PUSH    DE              ;Save variable address
```

```
  022CA 7E              LD      A,(HL)          ;A = step-SGN
  022CB 23              INC     HL              ;Pointer + 1
  022CC F5              PUSH    AF              ;Save step-SGN
  022CD D5              PUSH    DE              ;Save variable address
  022CE 7E              LD      A,(HL)          ;Get variable type
  022CF 23              INC     HL              ;Pointer + 1
  022D0 B7              OR      A               ;Loop variable type INT ?
  022D1 FAEA22          JP      M,22EAH         ;Yes: continue at 22EAH

; NEXT for SNG-variable

  022D4 CDB109          CALL    09B1H           ;X = BCDE + (HL)
                                                ;(Step value into X)
  022D7 E3              EX      (SP),HL         ;Save pointer, HL = VARPTR
  022D8 E5              PUSH    HL              ;Save address of loop var.
  022D9 CD0B07          CALL    070BH           ;X = X + (HL), add step value
                                                ;to loop variable
  022DC E1              POP     HL              ;Restore address
  022DD CDCB09          CALL    09CBH           ;(HL) = X (copy new value
                                                ;into loop variable
  022E0 E1              POP     HL              ;Restore pointer (now on end
                                                ;value)
  022E1 CDC209          CALL    09C2H           ;BCDE = (HL) = end value
  022E4 E5              PUSH    HL              ;Save pointer
  022E5 CD0C0A          CALL    0A0CH           ;CP X,BCDE (compare loop
                                                ;variable with end value)
  022E8 1829            JR      2313H           ;Continue at 2313H

; NEXT for INT-variable

  022EA 23              INC     HL              ;The first 4 bytes of the FOR
  022EB 23              INC     HL              ;stack are not used
  022EC 23              INC     HL
  022ED 23              INC     HL
  022EE 4E              LD      C,(HL)          ;BC = step value
  022EF 23              INC     HL
  022F0 46              LD      B,(HL)
  022F1 23              INC     HL
  022F2 E3              EX      (SP),HL         ;Save pointer, HL = address
  022F3 5E              LD      E,(HL)          ;DE = loop value
  022F4 23              INC     HL
  022F5 56              LD      D,(HL)
  022F6 E5              PUSH    HL              ;Save address + 1
  022F7 69              LD      L,C             ;HL = step value
  022F8 60              LD      H,B
  022F9 CDD20B          CALL    0BD2H           ;X = HL = HL + DE (add step
                                                ;value to loop variable)
  022FC 3AAF40          LD      A,(40AFH)       ;Overflow ?
  022FF FE04            CP      04H             ;VT changed to SNG ?
  02301 CAB207          JP      Z,07B2H         ;Yes: ?OV Error
                                                ;No:
  02304 EB              EX      DE,HL           ;DE = new loop value
  02305 E1              POP     HL              ;Restore address + 1
  02306 72              LD      (HL),D          ;Save new loop variable value
  02307 2B              DEC     HL
  02308 73              LD      (HL),E
  02309 E1              POP     HL              ;Restore pointer
  0230A D5              PUSH    DE              ;Save loop value
  0230B 5E              LD      E,(HL)          ;DE = end value
  0230C 23              INC     HL
  0230D 56              LD      D,(HL)
  0230E 23              INC     HL
  0230F E3              EX      (SP),HL         ;Save pointer, HL = loop value
  02310 CD390A          CALL    0A39H           ;CP HL,DE

  02313 E1              POP     HL              ;Restore pointer
  02314 C1              POP     BC              ;B = step-SGN
                                                ;B = 01H: End loop if
                                                ;         loop val. > end val.
                                                ;B = FFH: End loop if
                                                ;         loop val. < end val.
```

```
                                                   ;A = FFH: loop val. < end val.
                                                   ;A = 00H: loop val. = end val.
                                                   ;A = 01H: loop val. > end val.
  02315 90               SUB     B                 ;Loop ended ?
  02316 CDC209           CALL    09C2H             ;BCDE = (HL)
                                                   ;(BC = LP on start of loop
                                                   ;DE = line number)
  02319 2809             JR      Z,2324H           ;Yes: continue at 2324H

; Loop continues

  0231B EB               EX      DE,HL             ;HL = line number
  0231C 22A240           LD      (40A2H),HL        ;Save as current line number
  0231F 69               LD      L,C               ;HL = line pointer
  02320 60               LD      H,B
  02321 C31A1D           JP      1D1AH             ;Rebuild FOR stack and continue
                                                   ;program execution at (HL)
; Loop ends

  02324 F9               LD      SP,HL             ;SP = end of stack (current
                                                   ;FOR stack deleted)
  02325 22E840           LD      (40E8H),HL        ;Save new SP in system RAM
  02328 2ADF40           LD      HL,(40DFH)        ;HL = PTP
  0232B 7E               LD      A,(HL)            ;A = next character
  0232C FE2C             CP      ','               ;Is it a comma ?
  0232E C21E1D           JP      NZ,1D1EH          ;No: continue program at (HL)
                                                   ;Yes:
  02331 D7               RST     10H               ;Increment PTP to next variable
  02332 CDB922           CALL    22B9H             ;and process next variable


; Process expression in parentheses and store result in X

  02335 CF               RST     08H               ;Next character must be
  02336 28               DEFB    '('               ;a '('

; Process expression and store result in X

  02337 2B               DEC     HL                ;PTP - 1
  02338 1600             LD      D,00H             ;Priority flag = 0
  0233A D5               PUSH    DE                ;Save flag
  0233B 0E01             LD      C,01H             ;Memory test: still 2 bytes
  0233D CD6319           CALL    1963H             ;free?
  02340 CD9F24           CALL    249FH             ;Get 1st argument
  02343 22F340           LD      (40F3H),HL        ;Save PTP in system RAM
  02346 2AF340           LD      HL,(40F3H)        ;Restore PTP
  02349 C1               POP     BC                ;Restore priority flag
  0234A 7E               LD      A,(HL)            ;A = token following argument
                                                   ;(operator code)
  0234B 1600             LD      D,00H             ;Operator flag = 0
  0234D D6D4             SUB     0D4H              ;Comparison operator ?
                                                   ;( < , = , > )
  0234F 3813             JR      C,2364H           ;No: continue at 2364H

  02351 FE03             CP      03H               ;Token value between
                                                   ;D4H and D7H ?
  02353 300F             JR      NC,2364H          ;No: continue at 2364H

; Process operators for compare ( < , = , > , =< , => , <= , >= , <> )

  02355 FE01             CP      01H               ;C-flag = 1 in case of '<'
  02357 17               RLA                       ;'<' : A = 01 (bit 0 set)
                                                   ;'=' : A = 02 (bit 1 set)
                                                   ;'>' : A = 04 (bit 2 set)
  02358 AA               XOR     D                 ;Xor A with last operator.
                                                   ;A = 00 in case the same
                                                   ;operator was indicated twice
  02359 BA               CP      D                 ;C-flag = 1 in case A = 00H !!
  0235A 57               LD      D,A               ;D = new operator code
                                                   ;Twice the same operator ?
  0235B DA9719           JP      C,1997H           ;Yes: ?SN Error
```

```
  0235E 22D840          LD      (40D8H),HL      ;Save PTP
  02361 D7              RST     10H             ;Get next operator
  02362 18E9            JR      234DH           ;Another comparison operator ?

; No operator for compare found

  02364 7A              LD      A,D             ;A = operator flag
  02365 B7              OR      A               ;Comparison operators found ?
  02366 C2EC23          JP      NZ,23ECH        ;Yes: continue at 23ECH
                                                ;No:
  02369 7E              LD      A,(HL)          ;A = operator code
  0236A 22D840          LD      (40D8H),HL      ;Save PTP
  0236D D6CD            SUB     0CDH            ;Code found ?
  0236F D8              RET     C               ;No: done

  02370 FE07            CP      07H             ;Valid code ( + , - , * , / ,
                                                ;arrow up , AND , OR ) ?
  02372 D0              RET     NC              ;No: done

  02373 5F              LD      E,A             ;DE = code offset
                                                ;(0 = '+', 6 = 'OR')
  02374 3AAF40          LD      A,(40AFH)       ;A = VT of 1st argument
  02377 D603            SUB     03H             ;String argument ?
  02379 B3              OR      E               ;And code = '+' ?
  0237A CA8F29          JP      Z,298FH         ;Yes: string addition at 298FH

; Priority test

  0237D 219A18          LD      HL,189AH        ;HL -> priority table
  02380 19              ADD     HL,DE           ;Add offset
  02381 78              LD      A,B             ;A = old priority
                                                ;(at first = 0 !)
  02382 56              LD      D,(HL)          ;D = current priority
  02383 BA              CP      D               ;Compare both priorities
                                                ;Old priority was higher ?
  02384 D0              RET     NC              ;Yes: first calculate
                                                ;intermediate result
  02385 C5              PUSH    BC              ;Save old priority
  02386 014623          LD      BC,2346H        ;Set new RET-address
  02389 C5              PUSH    BC              ;to 2346H
  0238A 7A              LD      A,D             ;A = new priority
  0238B FE7F            CP      7FH             ;Power calculation (^) ?
  0238D CAD423          JP      Z,23D4H         ;Yes: continue at 23D4H

  02390 FE51            CP      51H             ;boolean operator (AND, OR) ?
  02392 DAE123          JP      C,23E1H         ;Yes: continue at 23E1H

; Process operators '+', '-', '*' and '/'
; Store 1st argument or intermediate result on stack

  02395 212141          LD      HL,4121H        ;HL -> X
  02398 B7              OR      A               ;C-flag = 0
  02399 3AAF40          LD      A,(40AFH)       ;A = VT
  0239C 3D              DEC     A               ;- 3
  0239D 3D              DEC     A
  0239E 3D              DEC     A               ;Type of X = STR ?
  0239F CAF60A          JP      Z,0AF6H         ;Yes: ?TM Error (for strings
                                                ;only '+' is allowed and this
                                                ;operator has already been
                                                ;checked)
  023A2 4E              LD      C,(HL)          ;BC = INT value
  023A3 23              INC     HL
  023A4 46              LD      B,(HL)
  023A5 C5              PUSH    BC              ;INT value on stack
                                                ;Type of X = INT ?
  023A6 FAC523          JP      M,23C5H         ;Yes: done

  023A9 23              INC     HL              ;BC = MSBs of SNG value
  023AA 4E              LD      C,(HL)
  023AB 23              INC     HL
```

```
  023AC 46              LD      B,(HL)
  023AD C5              PUSH    BC              ;MSBs of SNG-value on stack
  023AE F5              PUSH    AF              ;Save VT - 3
  023AF B7              OR      A               ;Type of X = SNG ?
  023B0 E2C423          JP      PO,23C4H        ;Yes: done

  023B3 F1              POP     AF              ;Restore VT - 3
  023B4 23              INC     HL
  023B5 3803            JR      C,23BAH         ;Jump is never executed
  023B7 211D41          LD      HL,411DH        ;Save LSBs of DBL value
  023BA 4E              LD      C,(HL)          ;on stack
  023BB 23              INC     HL
  023BC 46              LD      B,(HL)
  023BD 23              INC     HL
  023BE C5              PUSH    BC
  023BF 4E              LD      C,(HL)
  023C0 23              INC     HL
  023C1 46              LD      B,(HL)
  023C2 C5              PUSH    BC
  023C3 06F1            LD      B,0F1H          ;--

* 023C4   F1            POP     AF              ;Restore VT - 3 (with INT- or
                                                ;SNG-value)
  023C5 C603            ADD     A,03H           ;A = VT
  023C7 4B              LD      C,E             ;C = offset of operator code
  023C8 47              LD      B,A             ;B = VT
  023C9 C5              PUSH    BC              ;Save both on stack
  023CA 010624          LD      BC,2406H        ;Set new RET-address
  023CD C5              PUSH    BC              ;to 2406H
  023CE 2AD840          LD      HL,(40D8H)      ;Restore PTP
  023D1 C33A23          JP      233AH           ;Process next argument

  023D4 CDB10A          CALL    0AB1H           ;X = CSNG (X)
  023D7 CDA409          CALL    09A4H           ;(SP) = X
  023DA 01F213          LD      BC,13F2H        ;BC -> power function
  023DD 167F            LD      D,7FH           ;D = priority code
  023DF 18EC            JR      23CDH           ;Get 2nd argument and
                                                ;execute function

; Process logical expression

  023E1 D5              PUSH    DE              ;Save priority code
  023E2 CD7F0A          CALL    0A7FH           ;HL = X = CINT(X)
  023E5 D1              POP     DE              ;Restore priority code
  023E6 E5              PUSH    HL              ;Save 1st argument
  023E7 01E925          LD      BC,25E9H        ;BC -> AND/OR routine
  023EA 18E1            JR      23CDH           ;Get 2nd argument and
                                                ;process logical expression

; Process compare operators

  023EC 78              LD      A,B             ;A = priority code
  023ED FE64            CP      64H             ;Last operator of higher
                                                ;priority?
  023EF D0              RET     NC              ;Yes: compute intermediate
                                                ;result
  023F0 C5              PUSH    BC              ;No: save priority code
  023F1 D5              PUSH    DE              ;Save operator code
  023F2 110464          LD      DE,6404H        ;D = current priority
                                                ;E = offset for basic
                                                ;arithmatic (see table
                                                ;at 18ABH)
  023F5 21B825          LD      HL,25B8H        ;Set new RET-address
  023F8 E5              PUSH    HL              ;to 25B8H
  023F9 E7              RST     20H             ;TSTTYP
                                                ;STR type?
  023FA C29523          JP      NZ,2395H        ;No: continue at 2395H
                                                ;Yes:
  023FD 2A2141          LD      HL,(4121H)      ;HL -> string vector
  02400 E5              PUSH    HL              ;Save pointer
  02401 018C25          LD      BC,258CH        ;Set RET-address to 258CH
```

```
                                                ;(string compare)
  02404 18C7          JR      23CDH             ;Get 2nd argument and execute
                                                ;function

; Compute intermediate result
; 1st value: on stack, 2nd value: in X

  02406 C1            POP     BC                ;Restore VT of 1st argument
  02407 79            LD      A,C               ;A = operator code offset
  02408 32B040        LD      (40B0H),A         ;Save in system RAM
  0240B 78            LD      A,B               ;A = VT
  0240C FE08          CP      08H               ;1st argument in DBL format ?
  0240E 2828          JR      Z,2438H           ;Yes: continue at 2438H

  02410 3AAF40        LD      A,(40AFH)         ;A = VT of 2nd argument
  02413 FE08          CP      08H               ;2nd argument in DBL format ?
  02415 CA6024        JP      Z,2460H           ;Yes: continue at 2460H

  02418 57            LD      D,A               ;D = VT of 2nd argument
  02419 78            LD      A,B               ;A = VT of 1st argument
  0241A FE04          CP      04H               ;1st argument in SNG format ?
  0241C CA7224        JP      Z,2472H           ;Yes: continue at 2472H

  0241F 7A            LD      A,D               ;A = VT of 2nd argument
  02420 FE03          CP      03H               ;2nd argument in STR format ?
  02422 CAF60A        JP      Z,0AF6H           ;Yes: ?TM Error
                                                ;2nd argument in SNG format ?
  02425 D27C24        JP      NC,247CH          ;Yes: continue at 247CH

; 1st or 2nd argument in INT format

  02428 21BF18        LD      HL,18BFH          ;HL -> jump table for basic
                                                ;arithmatic (INT)
  0242B 0600          LD      B,00H             ;BC = operator offset
  0242D 09            ADD     HL,BC             ;Add offset twice
  0242E 09            ADD     HL,BC             ;(2 bytes address)
  0242F 4E            LD      C,(HL)            ;BC = address
  02430 23            INC     HL
  02431 46            LD      B,(HL)
  02432 D1            POP     DE                ;DE = 1st argument
  02433 2A2141        LD      HL,(4121H)        ;HL = 2nd argument
  02436 C5            PUSH    BC                ;Put address on stack
  02437 C9            RET                       ;and execute routine

; 1st argument in DBL format

  02438 CDDB0A        CALL    0ADBH             ;X = CDBL (X) (convert 2nd
                                                ;argument to DBL)
  0243B CDFC09        CALL    09FCH             ;Y = X
  0243E E1            POP     HL                ;Get 1st argument from stack
  0243F 221F41        LD      (411FH),HL        ;and load into X
  02442 E1            POP     HL
  02443 221D41        LD      (411DH),HL
  02446 C1            POP     BC
  02447 D1            POP     DE
  02448 CDB409        CALL    09B4H             ;X = BCDE
  0244B CDDB0A        CALL    0ADBH             ;X = CDBL (X) (convert 1st
                                                ;argument to DBL)
  0244E 21AB18        LD      HL,18ABH          ;HL -> jump table for basic
                                                ;arithmatic (DBL)
  02451 3AB040        LD      A,(40B0H)         ;A = operator code offset
  02454 07            RLCA                      ;* 2 (2 bytes offset)
  02455 C5            PUSH    BC                ;Save BC
  02456 4F            LD      C,A               ;BC = table offset
  02457 0600          LD      B,00H
  02459 09            ADD     HL,BC             ;Add offset
  0245A C1            POP     BC                ;Restore BC
  0245B 7E            LD      A,(HL)            ;HL -> routine
  0245C 23            INC     HL
  0245D 66            LD      H,(HL)
  0245E 6F            LD      L,A
```

```
  0245F E9              JP      (HL)            ;Execute routine

; 2nd argument in DBL format

  02460 C5              PUSH    BC              ;Save VT offset
  02461 CDFC09          CALL    09FCH           ;Y = X (2nd argument to X)
  02464 F1              POP     AF              ;Restore VT
  02465 32AF40          LD      (40AFH),A       ;Save VT in system RAM
  02468 FE04            CP      04H             ;1st argument in SNG format ?
  0246A 28DA            JR      Z,2446H         ;Yes: get 1st argument from
                                                ;stack and convert to DBL
  0246C E1              POP     HL              ;No: 1st argument is an INT:
                                                ;Get 1st argument
  0246D 222141          LD      (4121H),HL      ;Put it into X
  02470 18D9            JR      244BH           ;and convert it to DBL

; 1st argument in SNG format

  02472 CDB10A          CALL    0AB1H           ;X = CSNG (X) (convert 2nd
                                                ;argument to SNG)
  02475 C1              POP     BC              ;Get 1st argument from stack
  02476 D1              POP     DE
  02477 21B518          LD      HL,18B5H        ;HL -> jump table
  0247A 18D5            JR      2451H           ;Execute function

; 2nd argument in SNG format, 1st argument in INT format

  0247C E1              POP     HL              ;Restore 1st argument
  0247D CDA409          CALL    09A4H           ;(SP) = X (save 2nd argument
                                                ;on stack)
  02480 CDCF0A          CALL    0ACFH           ;X = CSNG (HL)
  02483 CDBF09          CALL    09BFH           ;BCDE = X = 1st argument
  02486 E1              POP     HL              ;1st argument from stack
  02487 222341          LD      (4123H),HL      ;Put it in X
  0248A E1              POP     HL
  0248B 222141          LD      (4121H),HL
  0248E 18E7            JR      2477H           ;Continue at 2477H


; IDIV: X = DE / HL
; Divides two integer numbers with the result in SNG format
;
; I: DE = Dividend
;    HL = Divisor
; O: X = Quotient

  02490 E5              PUSH    HL              ;Save divisor
  02491 EB              EX      DE,HL           ;HL = dividend
  02492 CDCF0A          CALL    0ACFH           ;X = CSNG (DE) (X = dividend)
  02495 E1              POP     HL              ;Restore dividend
  02496 CDA409          CALL    09A4H           ;(SP) = X (dividend to stack)
  02499 CDCF0A          CALL    0ACFH           ;X = CSNG (HL) (X = divisor)
  0249C C3A008          JP      08A0H           ;X = (SP) / X (execute SDIV)


; Decode argument at (PTP) and store result in X

  0249F D7              RST     10H             ;Operand indicated ?
  024A0 1E28            LD      E,28H           ;E = error code for ?MO Error
  024A2 CAA219          JP      Z,19A2H         ;No: ?MO Error
                                                ;Yes: number indicated ?
  024A5 DA6C0E          JP      C,0E6CH         ;Yes: decode number and store
                                                ;result in X
                                                ;No:
  024A8 CD3D1E          CALL    1E3DH           ;Variable indicated ?
  024AB D24025          JP      NC,2540H        ;Yes: continue at 2540H

  024AE FECD            CP      0CDH            ;'+' token ? (positive sign)
  024B0 28ED            JR      Z,249FH         ;Yes: skip it, get next
                                                ;character
  024B2 FE2E            CP      2EH             ;'.' ? (decimal point)
```

```
  024B4 CA6C0E          JP      Z,0E6CH         ;Yes: decode number

  024B7 FECE            CP      0CEH            ;'-' token ? (negative sign)
  024B9 CA3225          JP      Z,2532H         ;Yes: continue at 2532H

  024BC FE22            CP      22H             ;'"' ? (string constant)
  024BE CA6628          JP      Z,2866H         ;Yes: continue at 2866H

  024C1 FECB            CP      0CBH            ;'NOT' token ?
  024C3 CAC425          JP      Z,25C4H         ;Yes: continue at 25C4H

  024C6 FE26            CP      26H             ;'&' ? (hexadecimal constant)
  024C8 CAE334          JP      Z,34E3H         ;Yes: continue at 34E3H

  024CB FEC3            CP      0C3H            ;'ERR' token ?
  024CD 200A            JR      NZ,24D9H        ;No: continue at 24D9H



; X = ERR
; -------

  024CF D7              RST     10H             ;Adjust PTP
  024D0 3A9A40          LD      A,(409AH)       ;A = last error code
  024D3 E5              PUSH    HL              ;Save PTP
  024D4 CDF827          CALL    27F8H           ;Save A to X as INT
  024D7 E1              POP     HL              ;Restore PTP
  024D8 C9              RET


; Not ERR

  024D9 FEC2            CP      0C2H            ;'ERL' token ?
  024DB 200A            JR      NZ,24E7H        ;No: continue at 24E7H


; X = ERL
; -------

  024DD D7              RST     10H             ;Adjust PTP
  024DE E5              PUSH    HL              ;Save PTP
  024DF 2AEA40          LD      HL,(40EAH)      ;HL = LN of last error line
  024E2 CD660C          CALL    0C66H           ;X = HL (SNG) (no sign)
  024E5 E1              POP     HL              ;Restore PTP
  024E6 C9              RET


; not ERL

  024E7 FEC0            CP      0C0H            ;'VARPTR' toekn ?
  024E9 2014            JR      NZ,24FFH        ;No: continue at 24FFH


; X = VARPTR()
; ------------

  024EB D7              RST     10H             ;Adjust PTP
  024EC CF              RST     08H             ;Next character must be
  024ED 28              DEFB    '('             ;a '('
  024EE CD0D26          CALL    260DH           ;DE = address of indicated
                                                ;variable. DE = 0000H in case
                                                ;variable not found
  024F1 CF              RST     08H             ;Next character must be
  024F2 29              DEFB    ')'             ;a ')'
  024F3 E5              PUSH    HL              ;Save PTP
  024F4 EB              EX      DE,HL           ;HL -> variable
  024F5 7C              LD      A,H             ;Null pointer ?
  024F6 B5              OR      L
  024F7 CA4A1E          JP      Z,1E4AH         ;Yes: ?FC Error
                                                ;No:
  024FA CD9A0A          CALL    0A9AH           ;X = HL (INT)
```

```
  024FD E1            POP    HL          ;Restore PTP
  024FE C9            RET
```

; Not VARPTR

```
  024FF FEC1          CP     0C1H        ;'USR" token ?
  02501 C37A3F        JP     3F7AH       ;Intercept Colour BASIC token
                                         ;and continue at 27FEH in case
                                         ;'USR' token found

  02504 FEC5          CP     0C5H        ;'INSTR' token ?
  02506 CA9D41        JP     Z,419DH     ;Yes: continue at 41D9H

  02509 FEC8          CP     0C8H        ;'MEM' token ?
  0250B CAC927        JP     Z,27C9H     ;Yes: continue at 27C9H

  0250E FEC7          CP     0C7H        ;'TIME$' token ?
  02510 CA7641        JP     Z,4176H     ;Yes: continue at 4176H

  02513 FEC6          CP     0C6H        ;'CHECK' token ?
  02515 CA3201        JP     Z,0132H     ;Yes: continue at 0132H

  02518 FEC9          CP     0C9H        ;'INKEY$' token ?
  0251A CA9D01        JP     Z,019DH     ;Yes: continue at 01D9H

  0251D FEC4          CP     0C4H        ;'STRING$' token ?
  0251F CA2F2A        JP     Z,2A2FH     ;Yes: continue at 2A2FH

  02522 FEBE          CP     0BEH        ;'FN' token ?
  02524 CA5541        JP     Z,4155H     ;Yes: continue at 4155H

  02527 D6D7          SUB    0D7H        ;Function token found ?
  02529 D24E25        JP     NC,254EH    ;Yes: continue at 254EH
                                         ;No:
  0252C CD3523        CALL   2335H       ;Since all other possibilities
                                         ;have been tested, there can
                                         ;only be an expresssion
                                         ;enclosed in paranthesis
                                         ;at (PTP)
  0252F CF            RST    08H         ;Next character must be
  02530 29            DEFB   ')'         ;a ')'
  02531 C9            RET
```

; Process negative sign

```
  02532 167D          LD     D,7DH       ;D = priority code (next
                                         ;priority after power function)
  02534 CD3A23        CALL   233AH       ;Argument to X
  02537 2AF340        LD     HL,(40F3H)  ;Get PTP
  0253A E5            PUSH   HL          ;And save it
  0253B CD7B09        CALL   097BH       ;Negate result (type conform)
  0253E E1            POP    HL          ;Restore PTP
  0253F C9            RET
```

; Process variable as operand

```
  02540 CD0D26        CALL   260DH       ;Get address of variable
  02543 E5            PUSH   HL          ;Save PTP
  02544 EB            EX     DE,HL       ;HL = address
  02545 222141        LD     (4121H),HL  ;Save address
  02548 E7            RST    20H         ;TSTTYP
                                         ;STR type ?
  02549 C4F709        CALL   NZ,09F7H    ;No: X = (HL) (type conform)
  0254C E1            POP    HL          ;Restore PTP
  0254D C9            RET
```

; Function token found

```
; A = 00H for SNG, 01H for INT,..., 22H for RIGHT$ and 24H for MID$
; (see keyword and address table at 1608H)

  0254E 0600            LD      B,00H           ;MSB of offset = 00H
  02550 07              RLCA                    ;A = A * 2 (2 bytes address)
  02551 4F              LD      C,A             ;BC = offset of address table
  02552 C5              PUSH    BC              ;Save offset
  02553 D7              RST     10H             ;Adjust PTP
  02554 79              LD      A,C             ;A = offset LSB
  02555 FE41            CP      41H             ;LSB > 41H ?
  02557 3816            JR      C,256FH         ;No: continue at 256FH


; Process 1st argument of LEFT$, RIGHT$ or MID$

  02559 CD3523          CALL    2335H           ;Process expression
  0255C CF              RST     08H             ;Next character must be
  0255D 2C              DEFB    ','             ;a ','
                                                ;1st argument not a string ?
  0255E CDF40A          CALL    0AF4H           ;Yes: ?TM Error
  02561 EB              EX      DE,HL           ;DE = PTP
  02562 2A2141          LD      HL,(4121H)      ;HL -> 1st argument
  02565 E3              EX      (SP),HL         ;Save pointer
  02566 E5              PUSH    HL              ;Save offset to stack
  02567 EB              EX      DE,HL           ;HL = PTP
  02568 CD1C2B          CALL    2B1CH           ;DE = 2nd argument
  0256B EB              EX      DE,HL           ;DE = PTP, HL = 2nd argument
  0256C E3              EX      (SP),HL         ;Save 2nd argument, HL = offset
  0256D 1814            JR      2583H           ;Continue at 2583H


; Get function argument and execute function

  0256F CD2C25          CALL    252CH           ;Process expression
  02572 E3              EX      (SP),HL         ;Save PTP, HL = offset
  02573 7D              LD      A,L             ;A = LSB of offset
  02574 FE0C            CP      0CH             ;LSB < 0CH ? (SNG/INT/ABS/FRE/
                                                ;INP/POS)
  02576 3807            JR      C,257FH         ;Yes: execute function
                                                ;No:
  02578 FE1B            CP      1BH             ;LSB < 1BH ? (SQR/RND/LOG/EXP/
                                                ;COS/SIN/TAN/ATN)
  0257A E5              PUSH    HL              ;Save offset
  0257B DCB10A          CALL    C,0AB1H         ;Yes: X = CSNG (X)
  0257E E1              POP     HL              ;Restore offset
  0257F 113E25          LD      DE,253EH        ;Set new RET-address
  02582 D5              PUSH    DE              ;to 253EH
  02583 010816          LD      BC,1608H        ;BC -> jump table
  02586 09              ADD     HL,BC           ;Add offset
  02587 4E              LD      C,(HL)          ;Load jump address in HL
  02588 23              INC     HL
  02589 66              LD      H,(HL)
  0258A 69              LD      L,C
  0258B E9              JP      (HL)            ;Execute function


; String compare
;
; I: (SP-2) = RET addres to 25B8H
;    (SP)   = Vector address of 1st string
;     X     = Vector address of 2nd string
; O:  A     = FFH : 1st string < 2nd string
;           = 00H : 1st string = 2nd string
;           = 01H : 1st string > 2nd string

  0258C CDD729          CALL    29D7H           ;Test 2nd argument if STR
                                                ;format, HL -> vector of
                                                ;2nd string
  0258F 7E              LD      A,(HL)          ;A = LEN (2nd string)
  02590 23              INC     HL              ;BC = address (2nd string)
  02591 4E              LD      C,(HL)
```

```
  02592 23              INC     HL
  02593 46              LD      B,(HL)
  02594 D1              POP     DE              ;Restore vector address of
                                                ;1st string in DE
  02595 C5              PUSH    BC              ;Save address and
  02596 F5              PUSH    AF              ;length of 2nd string
  02597 CDDE29          CALL    29DEH           ;HL -> vector of 2st string
  0259A D1              POP     DE              ;D = length of 2nd string
  0259B 5E              LD      E,(HL)          ;E = length of 1st string
  0259C 23              INC     HL              ;BC = address of 1st string
  0259D 4E              LD      C,(HL)
  0259E 23              INC     HL
  0259F 46              LD      B,(HL)
  025A0 E1              POP     HL              ;HL = address of 2nd string
  025A1 7B              LD      A,E             ;Both length counters zero ?
  025A2 B2              OR      D
  025A3 C8              RET     Z               ;Yes: done, return

  025A4 7A              LD      A,D             ;A = 2nd length counter
  025A5 D601            SUB     01H             ;Counter zero?
  025A7 D8              RET     C               ;Yes: done, return

  025A8 AF              XOR     A               ;A = 00H
  025A9 BB              CP      E               ;1st length counter zero ?
  025AA 3C              INC     A               ;A = 01H if so.
  025AB D0              RET     NC              ;Yes: done, return (A = 01H)

  025AC 15              DEC     D               ;2nd length counter - 1
  025AD 1D              DEC     E               ;1st length counter - 1
  025AE 0A              LD      A,(BC)          ;Compare next character of 1st
  025AF BE              CP      (HL)            ;string with character of 2nd
                                                ;string
  025B0 23              INC     HL              ;2nd pointer + 1
  025B1 03              INC     BC              ;1st pointer + 1
                                                ;Both characters identical ?
  025B2 28ED            JR      Z,25A1H         ;Yes: compare next characters

  025B4 3F              CCF                     ;Negate C-flag
  025B5 C36009          JP      0960H           ;A = FFH in case C-flag = 1
                                                ;A = 00H in case C-flag = 0

; Evaluate result of compare
;
; I: A = FFH : 1st argument < 2nd argument
;      = 00H : 1st argument = 2nd argument
;      = 01H : 1st argument > 2nd argument

  025B8 3C              INC     A               ;A = A + 1 (C-flag = 1 if A
                                                ;was FFH)
  025B9 8F              ADC     A,A             ;A = 01H : 1st arg. < 2nd arg.
                                                ;  = 02H : 1st arg. = 2nd arg.
                                                ;  = 04H : 1st arg. > 2nd arg.
  025BA C1              POP     BC              ;Restore compare operator code
                                                ;(see 2355H)
  025BB A0              AND     B               ;Set bits of operators done
  025BC C6FF            ADD     A,0FFH          ;A = A + FFH: C-flag = 1 if
                                                ;a condition was true (overflow
                                                ;in case A <> 0)
  025BE 9F              SBC     A,A             ;A = 00H = 0 in case false
                                                ;else A = FFH = -1
  025BF CD8D09          CALL    098DH           ;X = HL = A (with sign)
  025C2 1812            JR      25D6H           ;Back to 2346H


; NOT
; ---

  025C4 165A            LD      D,5AH           ;Priority code = 5AH (higher
                                                ;than AND and OR)
  025C6 CD3A23          CALL    233AH           ;Argument to X
  025C9 CD7F0A          CALL    0A7FH           ;HL = X = CINT (X)
```

```
  025CC 7D              LD      A,L             ;Negate result
  025CD 2F              CPL
  025CE 6F              LD      L,A
  025CF 7C              LD      A,H
  025D0 2F              CPL
  025D1 67              LD      H,A
  025D2 222141          LD      (4121H),HL      ;And write back
  025D5 C1              POP     BC              ;Remove RET-address
  025D6 C34623          JP      2346H           ;and back to 2346H


; RST 20H
; -------
; TSTTYP: Test current VT and set corresponding flags
;
; I: -
; O: A = VT - 3
;    INT: A = FFH, Z-flag = 0, C-flag = 1, S-flag = 1
;    STR: A = 00H, Z-flag = 1, C-flag = 1, S-flag = 0
;    SNG: A = 01H, Z-flag = 0, C-flag = 1, S-flag = 0
;    DBL: A = 05H, Z-flag = 0, C-flag = 0, S-flag = 0

  025D9 3AAF40          LD      A,(40AFH)       ;A = VT
  025DC FE08            CP      08H             ;DBL ?
  025DE 3005            JR      NC,25E5H        ;Yes: continue at 25E5H
                                                ;No:
  025E0 D603            SUB     03H             ;A = VT - 3
  025E2 B7              OR      A               ;Set flags
  025E3 37              SCF                     ;C-flag = 1
  025E4 C9              RET

  025E5 D603            SUB     03H             ;A = VT - 3
  025E7 B7              OR      A               ;Set flags
  025E8 C9              RET


; Process AND / OR

  025E9 C5              PUSH    BC              ;Save priority code
  025EA CD7F0A          CALL    0A7FH           ;HL = X = CINT (X) = 2nd arg.
  025ED F1              POP     AF              ;Restore priority code
  025EE D1              POP     DE              ;DE = 1st argument
  025EF 01FA27          LD      BC,27FAH        ;Set new RET-address
  025F2 C5              PUSH    BC              ;to 27FAH (X = HL (INT) )
  025F3 FE46            CP      46H             ;Priority code = 46H (OR) ?
  025F5 2006            JR      NZ,25FDH        ;No: execute AND at 25FDH


; OR
; --

  025F7 7B              LD      A,E             ;OR both arguments
  025F8 B5              OR      L
  025F9 6F              LD      L,A
  025FA 7C              LD      A,H
  025FB B2              OR      D
  025FC C9              RET                     ;Back to 27FAH


; AND
; ---

  025FD 7B              LD      A,E             ;AND both arguments
  025FE A5              AND     L
  025FF 6F              LD      L,A
  02600 7C              LD      A,H
  02601 A2              AND     D
  02602 C9              RET                     ;Back to 27FAH


; Return after DIM
```

```
  02603 2B                DEC     HL              ;PTP - 1
  02604 D7                RST     10H             ;End of command ?
  02605 C8                RET     Z               ;Yes: done, return
                                                  ;No:
  02606 CF                RST     08H             ;Next character must be
  02607 2C                DEFB    ','             ;a comma


; DIM statement
; -------------

  02608 010326            LD      BC,2603H        ;Set new return address

  0260B C5                PUSH    BC              ;to 2603H
  0260C F6AF              OR      0AFH            ;A <> 0 for DIM

; Find address of variable in (PTP) and create variable if it does not
; exist.
;
; I: PTP -> variable name
; O: DE -> searched variable (0000H if variable does not exist)

* 0260D   AF              XOR     A               ;A = 0 for address search
  0260E 32AE40            LD      (40AEH),A       ;Store flag
  02611 46                LD      B,(HL)          ;B = first character of name

  02612 CD3D1E            CALL    1E3DH           ;Is the character in (HL) a
                                                  ;upper case character ?
  02615 DA9719            JP      C,1997H         ;No: ?SN Error

  02618 AF                XOR     A               ;A = 00H
  02619 4F                LD      C,A             ;C = default second character
                                                  ;of variable name
  0261A D7                RST     10H             ;Second character present ?
  0261B 3805              JR      C,2622H         ;Jump when it is a number

  0261D CD3D1E            CALL    1E3DH           ;Upper case character present ?
  02620 3809              JR      C,262BH         ;No: keep variable name at
                                                  ;one character
  02622 4F                LD      C,A             ;C = second character of name
  02623 D7                RST     10H             ;Get next character
                                                  ;Is it a number ?
  02624 38FD              JR      C,2623H         ;Yes: get character

  02626 CD3D1E            CALL    1E3DH           ;Is it an upper case character ?
  02629 30F8              JR      NC,2623H        ;Yes: get character

  0262B 115226            LD      DE,2652H        ;Set new return address
  0262E D5                PUSH    DE              ;to 2652H

  0262F 1602              LD      D,02H           ;D = 2 (type code for INT)
  02631 FE25              CP      '%'             ;'%' (INT indentifier) found ?
  02633 C8                RET     Z               ;Yes: D = type code

  02634 14                INC     D               ;D = 3 (type code for STR)
  02635 FE24              CP      '$'             ;'$' (STR indentifier) found ?
  02637 C8                RET     Z               ;Yes: D = type code

  02638 14                INC     D               ;D = 4 (type code for SNG)
  02639 FE21              CP      '!'             ;'!' (SNG indentifier) found ?
  0263B C8                RET     Z               ;Yes: D = type code

  0263C 1608              LD      D,08H           ;D = 8 (type code for DBL)
  0263E FE23              CP      '#'             ;'#' (DBL indentifier) found ?
  02640 C8                RET     Z               ;Yes: D = type code

; No type code indicated. Get type code from DEF table

  02641 78                LD      A,B             ;A = first character of name
  02642 D641              SUB     41H             ;A = offset for type code table
```

```
  02644 E67F              AND     7FH              ;Clear highest bit

  02646 5F                LD      E,A              ;DE = offset
  02647 1600              LD      D,00H
  02649 E5                PUSH    HL               ;Save PTP
  0264A 210141            LD      HL,4101H         ;HL -> type code table
  0264D 19                ADD     HL,DE            ;Add offset
  0264E 56                LD      D,(HL)           ;Get type code
  0264F E1                POP     HL               ;Restore PTP
  02650 2B                DEC     HL               ;PTP - 1
  02651 C9                RET

; Search variable / create variable
; BC = 1st and 2nd character of variable name
; D = type code of variable

  02652 7A                LD      A,D              ;A = type code
  02653 32AF40            LD      (40AFH),A        ;Save as VT
  02656 D7                RST     10H              ;Skip spaces/LF etc
  02657 3ADC40            LD      A,(40DCH)        ;Array variables indicated ?
  0265A B7                OR      A                ;(see 1CA1H)
  0265B C26426            JP      NZ,2664H         ;No: create normal variable

  0265E 7E                LD      A,(HL)           ;A = next character following
                                                   ;variable name.
  0265F D628              SUB     28H              ;Is it a '('
  02661 CAE926            JP      Z,26E9H          ;Yes: process array variable

  02664 AF                XOR     A                ;A = 00H
  02665 32DC40            LD      (40DCH),A        ;Release array variable

  02668 E5                PUSH    HL               ;Save PTP
  02669 D5                PUSH    DE               ;Save type code

  0266A 2AF940            LD      HL,(40F9H)       ;HL -> start of variables
  0266D EB                EX      DE,HL            ;DE = HL
  0266E 2AFB40            LD      HL,(40FBH)       ;HL -> end of variables
  02671 DF                RST     18H              ;End reached ?
  02672 E1                POP     HL               ;Restore type code
  02673 2819              JR      Z,268EH          ;Yes: create new variable

  02675 1A                LD      A,(DE)           ;A = type code of variable
                                                   ;addressed by DE
  02676 6F                LD      L,A              ;L = type code
  02677 BC                CP      H                ;Is it the searched code ?
  02678 13                INC     DE               ;Pointer + 1
  02679 200B              JR      NZ,2686H         ;No: continue at 2686H

  0267B 1A                LD      A,(DE)           ;A = second character of name
  0267C B9                CP      C                ;Same as in searched name ?
  0267D 2007              JR      NZ,2686H         ;No: continue at 2686H

  0267F 13                INC     DE               :Pointer + 1
  02680 1A                LD      A,(DE)           ;A = first charachter of name
  02681 B8                CP      B                ;Same as in searched name ?
  02682 CACC26            JP      Z,26CCH          ;Yes: continue at 26CCH

  02685 3E13              LD      A,13H            ;--
* 02685   13              INC     DE               ;Pointer + 1
  02687 13                INC     DE               ;Pointer + 1
  02688 E5                PUSH    HL               ;Save type code
  02689 2600              LD      H,00H            ;HL = length of variable found
  0268B 19                ADD     HL,DE            ;Put HL on next variable
  0268C 18DF              JR      266DH            ;Repeat compare

; Variable not found

  0268E 7C                LD      A,H              ;A = type code
  0268F E1                POP     HL               ;Restore PTP
  02690 E3                EX      (SP),HL          ;Save PTP, HL = RET address
  02691 F5                PUSH    AF               ;Save type code
```

```
  02692 D5              PUSH    DE              ;Save pointer on end of
                                                ;variables
  02693 11F124          LD      DE,24F1H        ;Was routine CALLed by VARPTR
                                                ;function ?
  02696 DF              RST     18H             ;(RET address = 24F1H)
  02697 2836            JR      Z,26CFH         ;Yes: continue at 26CFH

  02699 114325          LD      DE,2543H        ;CALLed from 2540H ?
  0269C DF              RST     18H             ;(RET address = 2543H)
  0269D D1              POP     DE              ;Restore pointer
  0269E 2835            JR      Z,26D5H         ;Yes: continue at 26D5H

; Create new variable

  026A0 F1              POP     AF              ;Restore type code
  026A1 E3              EX      (SP),HL         ;Save RET address, restore PTP
  026A2 E5              PUSH    HL              ;Save PTP
  026A3 C5              PUSH    BC              ;Save variable name
  026A4 4F              LD      C,A             ;C = type code
  026A5 0600            LD      B,00H           ;B = 00H
  026A7 C5              PUSH    BC              ;Save type code
                                                ;Calculate total length of var
  026A8 03              INC     BC              ;BC + 1 for type code
  026A9 03              INC     BC              ;BC + 2 for variable name
  026AA 03              INC     BC
  026AB 2AFD40          LD      HL,(40FDH)      ;HL -> start of free space
  026AE E5              PUSH    HL              ;Save pointer
  026AF 09              ADD     HL,BC           ;+ length = new start of
                                                ;free space
  026B0 C1              POP     BC              ;Restore old start address
  026B1 E5              PUSH    HL              ;Save new start address
  026B2 CD5519          CALL    1955H           ;Test memory space and move
                                                ;memory to create space for
                                                ;new variable
  026B5 E1              POP     HL              ;Restore new start address
  026B6 22FD40          LD      (40FDH),HL      ;Save it in system RAM
  026B9 60              LD      H,B             ;HL = new end address of
  026BA 69              LD      L,C             ;memory with single variables
  026BB 22FB40          LD      (40FBH),HL      ;Save it in system RAM
  026BE 2B              DEC     HL              ;Pointer - 1
  026BF 3600            LD      (HL),00H        ;Clear memory space for new
                                                ;variable
  026C1 DF              RST     18H             ;Done ?
  026C2 20FA            JR      NZ,26BEH        ;No: continue clear

  026C4 D1              POP     DE              ;Restore type code
                                                ;HL -> new variable
  026C5 73              LD      (HL),E          ;Store type code
  026C6 23              INC     HL              ;Pointer + 1
  026C7 D1              POP     DE              ;Restore variable name
  026C8 73              LD      (HL),E          ;Store 2nd character
  026C9 23              INC     HL              ;Pointer + 1
  026CA 72              LD      (HL),D          ;Store 1st character
  026CB EB              EX      DE,HL           ;DE -> mantissa of new variable
  026CC 13              INC     DE              ;DE -> new variable
  026CD E1              POP     HL              ;Restore PTP
  026CE C9              RET


; VARPTR function: variable not found

  026CF 57              LD      D,A             ;DE = 0000H
  026D0 5F              LD      E,A
  026D1 F1              POP     AF              ;Correct stack
  026D2 F1              POP     AF
  026D3 E3              EX      (SP),HL         ;Restore PTP, save RET address
  026D4 C9              RET                     ;Done


; Variable at expression evaluation (2337H) not found
; Set result to 0
```

```
  026D5 322441          LD      (4124H),A       ;Exp (X) = 0 -> X = 0
  026D8 C1              POP     BC              ;Correct stack
  026D9 67              LD      H,A             ;HL = 0000H
  026DA 6F              LD      L,A
  026DB 222141          LD      (4121H),HL      ;Set X (INT) to 0
  026DE E7              RST     20H             ;TSTTYP
                                                ;STR type ?
  026DF 2006            JR      NZ,26E7H        ;No: X is ok, continue at 26E7H
                                                ;Yes:
  026E1 212819          LD      HL,1928H        ;Vector address -> null string
  026E4 222141          LD      (4121H),HL      ;X = HL
  026E7 E1              POP     HL              ;Restore PTP
  026E8 C9              RET


; Array variable recognized
;
; BC = variable name
; HL = PTP
; A  = 00H

  026E9 E5              PUSH    HL              ;Save PTP
  026EA 2AAE40          LD      HL,(40AEH)      ;L = type code, H = DIM flag
  026ED E3              EX      (SP),HL         ;Save HL, restore PTP
  026EE 57              LD      D,A             ;D = 0 (number of dimensions)
  026EF D5              PUSH    DE              ;Save registers
  026F0 C5              PUSH    BC
  026F1 CD451E          CALL    1E45H           ;Get next dimension at (PTP)
  026F4 C1              POP     BC              ;Restore registers
  026F5 F1              POP     AF
  026F6 EB              EX      DE,HL           ;DE = PTP, HL = dim. value
  026F7 E3              EX      (SP),HL         ;Save dim. value, restore HL
  026F8 E5              PUSH    HL              ;Save HL again
  026F9 EB              EX      DE,HL           ;HL = PTP
  026FA 3C              INC     A               ;Dimension counter + 1
  026FB 57              LD      D,A             ;D = counter
  026FC 7E              LD      A,(HL)          ;Another dimension indicated ?
  026FD FE2C            CP      ','             ;Separator found ?
  026FF 28EE            JR      Z,26EFH         ;Yes: get next dimension
                                                ;No:
  02701 CF              RST     08H             ;Next character must be
  02702 29              DEFB    ')'             ;a ')'
  02703 22F340          LD      (40F3H),HL      ;Save PTP
  02706 E1              POP     HL              ;Restore type code and DIM flag
  02707 22AE40          LD      (40AEH),HL      ;and save in system RAM
  0270A D5              PUSH    DE              ;Save dimension counter
                                                ;The stack now contains the
                                                ;dimension values and the
                                                ;number of dimensions
  0270B 2AFB40          LD      HL,(40FBH)      ;HL -> start of array variable
                                                ;memory space
  0270E 3E19            LD      A,19H           ;--

* 0270F   19            ADD     HL,DE           ;Add array size. HL now
                                                ;points on the next array
  02710 EB              EX      DE,HL
  02711 2AFD40          LD      HL,(40FDH)      ;DE -> end of array variable
                                                ;memory space
  02714 EB              EX      DE,HL
  02715 DF              RST     18H             ;End of array vars reached ?
  02716 3AAF40          LD      A,(40AFH)       ;A = requested type code
  02719 2827            JR      Z,2742H         ;No: array variable not found.
                                                ;HL points to free memory space
                                                ;for the new array.

  0271B BE              CP      (HL)            ;Type code found ?
  0271C 23              INC     HL              ;Pointer + 1
  0271D 2008            JR      NZ,2727H        ;No: continue at 2727H

  0271F 7E              LD      A,(HL)          ;2nd character of var. name
```

```
  02720 B9                 CP      C               ;and compare it
  02721 23                 INC     HL              ;Pointer + 1
                                                   ;Characters the same ?
  02722 2004               JR      NZ,2728H        ;No: continue at 2728H

  02724 7E                 LD      A,(HL)          ;Compare 1st character
  02725 B8                 CP      B               ;The same ?
  02726 3E23               LD      A,23H           ;--

* 02726   23               INC     HL              ;Adjust pointer
  02728 23                 INC     HL              ;Pointer + 1
  02729 5E                 LD      E,(HL)          ;DE = total length
  0272A 23                 INC     HL
  0272B 56                 LD      D,(HL)
  0272C 23                 INC     HL
  0272D 20E0               JR      NZ,270FH        ;Array not found: check next

  0272F 3AAE40             LD      A,(40AEH)       ;New array to be created ?
  02732 B7                 OR      A
  02733 1E12               LD      E,12H           ;E = error code for ?DD Error
  02735 C2A219             JP      NZ,19A2H        ;Yes: the array already exists,
                                                   ;issue error.

  02738 F1                 POP     AF              ;No: A = number of dimensions
  02739 96                 SUB     (HL)            ;Does they match ?
  0273A CA9527             JP      Z,2795H         ;Yes: compute address of the
                                                   ;requested array element

  0273D 1E10               LD      E,10H           ;E = error code for ?BS Error
  0273F C3A219             JP      19A2H           ;Issue error


; Array not found
; HL -> free memory space for new array

  02742 77                 LD      (HL),A          ;Save type code
  02743 23                 INC     HL              ;Pointer + 1
  02744 5F                 LD      E,A             ;DE = size of array element
  02745 1600               LD      D,00H
  02747 F1                 POP     AF              ;A = number of dimensions
  02748 71                 LD      (HL),C          ;Save array name
  02749 23                 INC     HL
  0274A 70                 LD      (HL),B
  0274B 23                 INC     HL
  0274C 4F                 LD      C,A             ;C = dimension counter
  0274D CD6319             CALL    1963H           ;Sufficient memory available ?
  02750 23                 INC     HL              ;Pointer + 1
  02751 23                 INC     HL              ;Pointer + 1
  02752 22D840             LD      (40D8H),HL      ;Save pointer in system RAM
  02755 71                 LD      (HL),C          ;Save number of dimensions
  02756 23                 INC     HL              ;Pointer + 1
  02757 3AAE40             LD      A,(40AEH)       ;A = DIM flag
  0275A 17                 RLA                     ;If address search then
                                                   ;C-flag = 0.
                                                   ;If array create then
                                                   ;C-flag = 1
  0275B 79                 LD      A,C             ;A = dimension counter
  0275C 010B00             LD      BC,000BH        ;BC = default dimension value
                                                   ;Address search ?
  0275F 3002               JR      NC,2763H        ; Yes: continue at 2763H

  02761 C1                 POP     BC              ;Restore dimension value
  02762 03                 INC     BC              ;+ 1 for zero element
  02763 71                 LD      (HL),C          ;Save dimension value
  02764 23                 INC     HL
  02765 70                 LD      (HL),B
  02766 23                 INC     HL
  02767 F5                 PUSH    AF              ;Save counter
  02768 CDAA0B             CALL    0BAAH           ;DE = DE * BC: calculate the
                                                   ;total array size.
  0276B F1                 POP     AF              ;Restore counter
```

```
  0276C 3D          DEC   A              ;Counter - 1
                                         ;All dimensions done ?
  0276D 20ED        JR    NZ,275CH       ;No: next dimension
                                         ;Yes:
  0276F F5          PUSH  AF             ;Save 00H on stack
  02770 42          LD    B,D            ;BC = array size
  02771 4B          LD    C,E
  02772 EB          EX    DE,HL          ;DE -> start of new array
                                         ;HL = array size
  02773 19          ADD   HL,DE          ;HL -> new end of array
                                         ;variable memory space
                                         ;Memory overflow ?
  02774 38C7        JR    C,273DH        ;Yes: ?BS Error

  02776 CD6C19      CALL  196CH          ;Still free memory remaining ?
  02779 22FD40      LD    (40FDH),HL     ;Save new end address
  0277C 2B          DEC   HL             ;Pointer - 1
  0277D 3600        LD    (HL),00H       ;Clear array byte
  0277F DF          RST   18H            ;Start of array reached ?
  02780 20FA        JR    NZ,277CH       ;No: clear next byte

  02782 03          INC   BC             ;BC = number of required bytes
                                         ;+ 1
  02783 57          LD    D,A            ;D = 0
  02784 2AD840      LD    HL,(40D8H)     ;HL -> number of dimensions
  02787 5E          LD    E,(HL)         ;DE = number of dimensions
  02788 EB          EX    DE,HL          ;HL = dimension counter
                                         ;DE = pointer
  02789 29          ADD   HL,HL          ;HL = 2 * number of dimensions
                                         ;(Every dimension value is
                                         ;stored in 2 bytes)
  0278A 09          ADD   HL,BC          ;+ number of bytes required
                                         ;by the array elements
                                         ;= total size (in bytes)
  0278B EB          EX    DE,HL          ;DE = total size
  0278C 2B          DEC   HL             ;Pointer - 2
  0278D 2B          DEC   HL
  0278E 73          LD    (HL),E         ;Save total size
  0278F 23          INC   HL
  02790 72          LD    (HL),D
  02791 23          INC   HL
  02792 F1          POP   AF             ;Restore flags from stack
                                         ;Creation of new array (DIM) ?
  02793 3830        JR    C,27C5H        ;Yes: done.

; Compute address of required variable

  02795 47          LD    B,A            ;BC = 0000H
  02796 4F          LD    C,A
  02797 7E          LD    A,(HL)         ;A = dimension counter
  02798 23          INC   HL             ;Pointer + 1
  02799 16E1        LD    D,0E1H         ;--

* 0279A   E1        POP   HL             ;Restore pointer
  0279B 5E          LD    E,(HL)         ;DE = next dimension value
  0279C 23          INC   HL
  0279D 56          LD    D,(HL)
  0279E 23          INC   HL
  0279F E3          EX    (SP),HL        ;Save pointer, HL = requested
                                         ;dimension
  027A0 F5          PUSH  AF             ;Save counter
  027A1 DF          RST   18H            ;Requested dimension found ?
                                         ;Requested dim. > found dim. ?
  027A2 D23D27      JP    NC,273DH       ;Yes: ?BS Error
                                         ;No:
  027A5 CDAA0B      CALL  0BAAH          ;DE = DE * BC = found dimension
                                         ;* last value
  027A8 19          ADD   HL,DE          ;HL = requested dimension +
                                         ;found dimension * last value
  027A9 F1          POP   AF             ;Restore counter
  027AA 3D          DEC   A              ;Counter - 1
```

```
  027AB 44              LD      B,H             ;BC = array element counter
  027AC 4D              LD      C,L
  027AD 20EB            JR      NZ,279AH        ;Process next dimension

  027AF 3AAF40          LD      A,(40AFH)       ;A = type code
  027B2 44              LD      B,H             ;BC = index value of array
  027B3 4D              LD      C,L             ;element
  027B4 29              ADD     HL,HL           ;* 2 = offset for INT
  027B5 D604            SUB     04H             ;Typecode = INT or STR ?
  027B7 3804            JR      C,27BDH         ;Yes: continue at 27BDH
                                                ;No:
  027B9 29              ADD     HL,HL           ;* 2 = offset for SNG
                                                ;Type code = SNG ?
  027BA 2806            JR      Z,27C2H         ;Yes: continue at 27C2H
                                                ;No:
  027BC 29              ADD     HL,HL           ;* 2 = offset for DBL
  027BD B7              OR      A               ;Type code = INT or DBL ?
  027BE E2C227          JP      PO,27C2H        ;Yes: HL contains proper offset
                                                ;No:
  027C1 09              ADD     HL,BC           ;Add again: offset for STR

  027C2 C1              POP     BC              ;Start address array to BC
  027C3 09              ADD     HL,BC           ;+ offset = requested address
  027C4 EB              EX      DE,HL           ;DE = address
  027C5 2AF340          LD      HL,(40F3H)      ;HL = PTP
  027C8 C9              RET


; MEM = FRE (numerical variable )

  027C9 AF              XOR     A               ;A = 0
  027CA E5              PUSH    HL              ;Save PTP
  027CB 32AF40          LD      (40AFH),A       ;Set type code to 0
  027CE CDD427          CALL    27D4H           ;X = end of stack - start of
                                                ;free memory = number of
                                                ;free bytes
  027D1 E1              POP     HL              ;Restore PTP
  027D2 D7              RST     10H             ;Increase PTP
  027D3 C9              RET


; X = FRE ( arg )
; ---------------

  027D4 2AFD40          LD      HL,(40FDH)      ;HL = start of free memory
  027D7 EB              EX      DE,HL           ;DE = HL
  027D8 210000          LD      HL,0000H        ;HL = 0000H
  027DB 39              ADD     HL,SP           ;HL = 0 + SP = SP
  027DC E7              RST     20H             ;TSTTYP
                                                ;STR ? (number of free bytes
                                                ;in string space required)
  027DD 200D            JR      NZ,27ECH        ;No: continue at 27ECH
                                                ;Yes:
  027DF CDDA29          CALL    29DAH           ;Clear argument from string
                                                ;space
  027E2 CDE628          CALL    28E6H           ;Sort string space
  027E5 2AA040          LD      HL,(40A0H)      ;HL -> start of string space
  027E8 EB              EX      DE,HL           ;DE = HL
  027E9 2AD640          LD      HL,(40D6H)      ;HL -> last string in string
                                                ;space
  027EC 7D              LD      A,L             ;Compute the number of free
  027ED 93              SUB     E               ;bytes; result in HL
  027EE 6F              LD      L,A
  027EF 7C              LD      A,H
  027F0 9A              SBC     A,D
  027F1 67              LD      H,A
  027F2 C3660C          JP      0C66H           ;X = HL


; X = POS (dummy)
; ---------------
```

```
  027F5 3AA640          LD      A,(40A6H)       ;A = cursor position in line

  027F8 6F              LD      L,A             ;HL = A (without sign)
  027F9 AF              XOR     A
  027FA 67              LD      H,A
  027FB C39A0A          JP      0A9AH           ;X =  HL (INT)


; X = USR ( X )
; -------------

  027FE CDA941          CALL    41A9H           ;DOS
  02801 D7              RST     10H             ;Increase PTP
  02802 CD2C25          CALL    252CH           ;Process expression
  02805 E5              PUSH    HL              ;Save PTP
  02806 219008          LD      HL,0890H        ;Set new RET address
  02809 E5              PUSH    HL              ;to 0890H
  0280A 3AAF40          LD      A,(40AFH)       ;A = type code
  0280D F5              PUSH    AF              ;Save type code
  0280E FE03            CP      03H             ;STR argument?
  02810 CCDA29          CALL    Z,29DAH         ;Yes: remove argument from
                                                ;string space
  02813 F1              POP     AF              ;Restore type code
  02814 EB              EX      DE,HL           ;DE = 0890H (numerical
                                                ;argument)
                                                ;DE = vector address (string
                                                ;argument)
  02815 2A8E40          LD      HL,(408EH)      ;HL -> USR routine
  02818 E9              JP      (HL)            ;Execute routine


; Conversion of X into desired type
;
; I: A = desired type code
; (see 1F35H)

  02819 E5              PUSH    HL              ;Save HL
  0281A E607            AND     07H             ;A = offset for jump table
                                                ;INT: A = 2, STR A = 3
                                                ;SNG: A = 4, DBL A = 0
  0281C 21A118          LD      HL,18A1H        ;HL -> jump table for type
                                                ;conversion
  0281F 4F              LD      C,A             ;BC = offset
  02820 0600            LD      B,00H
  02822 09              ADD     HL,BC           ;Add offset to pointer
  02823 CD8625          CALL    2586H           ;Add offset again (2 bytes
                                                ;per address) and execute
                                                ;routine
  02826 E1              POP     HL              ;restore HL
  02827 C9              RET


; SUB for INPUT
; Test for ?ID Error
; (see 219AH)

  02828 E5              PUSH    HL              ;Save PTP
  02829 2AA240          LD      HL,(40A2H)      ;HL = current LN
  0282C 23              INC     HL              ;Current LN = 65535 ?
  0282D 7C              LD      A,H             ;(HL + 1 = 0000H)
  0282E B5              OR      L               ;(65535 = LN for command mode)
  0282F E1              POP     HL              ;Restore PTP
  02830 C0              RET     NZ              ;No: ok, return


; ?ID Error

  02831 1E16            LD      E,16H           ;E = error code for ?ID Error
  02833 C3A219          JP      19A2H           ;Issue error
```

```
; X = STR$ ( X )
; --------------

  02836 CDBD0F          CALL    0FBDH           ;Convert X into string from
                                                ;4130H onwards
  02839 CD6528          CALL    2865H           ;Take string constant
  0283C CDDA29          CALL    29DAH           ;and remove from string space
                                                ;HL -> string vector
  0283F 012B2A          LD      BC,2A2BH        ;Set new Return address
  02842 C5              PUSH    BC              ;to 2A2BH


; Insert new string into string space
;
; I: HL -> vector to new string (somewhere in memory)
; O: DE -> vector to new string (in string space)

  02843 7E              LD      A,(HL)          ;A = string length
  02844 23              INC     HL              ;Pointer + 1
  02845 E5              PUSH    HL              ;Save pointer
  02846 CDBF28          CALL    28BFH           ;Are there still A bytes free
                                                ;in string space ?
                                                ;Yes: DE -> 1st free byte for
                                                ;the new string
                                                ;No: ?OS Error
  02849 E1              POP     HL              ;Restore pointer
  0284A 4E              LD      C,(HL)          ;String address to BC
  0284B 23              INC     HL
  0284C 46              LD      B,(HL)
  0284D CD5A28          CALL    285AH           ;Store length and address of
                                                ;free memory (in string space)
                                                ;as last entry in string table
  02850 E5              PUSH    HL              ;Save vector address of new
                                                ;string space
  02851 6F              LD      L,A             ;L = string length
  02852 CDCE29          CALL    29CEH           ;Copy string from (BC) to (DE):
                                                ;Copy string to string space
  02855 D1              POP     DE              ;DE -> vector to new string
  02856 C9              RET


; Search for room in string space
;
; I: A = length of new string
; O: DE = start address in string space of the new string
;    HL = 40D3H = vector address of free string

  02857 CDBF28          CALL    28BFH           ;Still A bytes free in string
                                                ;space ?
                                                ;Yes: DE -> 1st free byte
                                                ;No: ?OS Error
  0285A 21D340          LD      HL,40D3H        ;HL -> vector of free string
  0285D E5              PUSH    HL              ;Save pointer
  0285E 77              LD      (HL),A          ;A = string length
  0285F 23              INC     HL
  02860 73              LD      (HL),E          ;Store address of free string
  02861 23              INC     HL              ;space memory
  02862 72              LD      (HL),D
  02863 E1              POP     HL              ;Retore vector address
  02864 C9              RET


; Process string constant at (HL) and store in string table
;
; I: HL -> string constant (terminated with '"' or 00H
; O: A = last string character
;    HL -> end of string
;    X = vector address of new string

  02865 2B              DEC     HL              ;Pointer - 1
```

```
  02866 0622          LD     B,22H           ;B = 1st delimiter char. ('"')
  02868 50            LD     D,B             ;D = 2nd delimiter character
  02869 E5            PUSH   HL              ;Save pointer
  0286A 0EFF          LD     C,0FFH          ;Set length counter to -1
  0286C 23            INC    HL              ;Pointer + 1
  0286D 7E            LD     A,(HL)          ;Get string character
  0286E 0C            INC    C               ;Counter + 1
  0286F B7            OR     A               ;End of string (00H) ?
  02870 2806          JR     Z,2878H         ;Yes: continue 2878H

  02872 BA            CP     D               ;2nd delimiter reached?
  02873 2803          JR     Z,2878H         ;Yes: end of string
                                             ;No:
  02875 B8            CP     B               ;Start delimiter reached ?
  02876 20F4          JR     NZ,286CH        ;No: get next character
                                             ;Yes:
  02878 FE22          CP     22H             ;String terminated by '"' ?
  0287A CC781D        CALL   Z,1D78H         ;Yes: increase pointer

  0287D E3            EX     (SP),HL         ;Save end pointer, HL = start
                                             ;pointer
  0287E 23            INC    HL              ;Pointer + 1
  0287F EB            EX     DE,HL           ;DE = start pointer
  02880 79            LD     A,C             ;C = length
  02881 CD5A28        CALL   285AH           ;Store length and pointer as
                                             ;last entry in string table
  02884 11D340        LD     DE,40D3H        ;DE = string vector
  02887 3ED5          LD     A,0D5H          ;--
* 02887   D5          PUSH   DE              ;Called from FN routine in DOS

  02889 2AB340        LD     HL,(40B3H)      ;HL = next free position in
                                             ;string table
  0288C 222141        LD     (4121H),HL      ;Save vector address in X
  0288F 3E03          LD     A,03H           ;Set VT to STR
  02891 32AF40        LD     (40AFH),A
  02894 CDD309        CALL   09D3H           ;Copy 3 bytes from (DE) to (HL)
                                             ;(copy length and address into
                                             ;string table)
  02897 11D640        LD     DE,40D6H        ;End of string table reached ?
  0289A DF            RST    18H             ;(HL = 40D6H)
  0289B 22B340        LD     (40B3H),HL      ;Save table pointer
  0289E E1            POP    HL              ;Restore end pointer
  0289F 7E            LD     A,(HL)          ;A = last character
  028A0 C0            RET    NZ              ;No: done, return
                                             ;Yes:
  028A1 1E1E          LD     E,1EH           ;E = error code for ?ST Error
  028A3 C3A219        JP     19A2H           ;Issue error


; Process and output text constant at (HL)
;
; I: HL -> string constant

  028A6 23            INC    HL              ;Pointer + 1
  028A7 CD6528        CALL   2865H           ;Copy string constant
  028AA CDDA29        CALL   29DAH           ;and remove it from string
                                             ;table
  028AD CDC409        CALL   09C4H           ;D = length, BC = start
                                             ;address of string
  028B0 14            INC    D               ;Counter + 1

  028B1 15            DEC    D               ;Counter - 1; counter = 0 ?
  028B2 C8            RET    Z               ;Yes: done, return

  028B3 0A            LD     A,(BC)          ;Get character
  028B4 CD2A03        CALL   032AH           ;Output character
  028B7 FE0D          CP     0DH             ;CR/LF ?
  028B9 CC0321        CALL   Z,2103H         ;Yes: call DOS
  028BC 03            INC    BC              ;Pointer + 1
  028BD 18F2          JR     28B1H           ;Next character
```

```
; Check for room in string space
; In case there are less than A bytes free in string space, the string space
; is sorted (garbage collection) and the routine is called again.
; If the string space is full a ?OS Error is generated
;
; I: A = number of required bytes
; O: DE -> A bytes free space in string space
;    HL -> start of string space

  028BF B7              OR      A               ;Z-flag = 0: 1st run
  028C0 0EF1            LD      C,0F1H          ;--

; Entry at 2nd call (after sorting)

* 028C0   F1            POP     AF              ;Restore AF (Z-flag is now 1 !)
  028C2 F5              PUSH    AF              ;Save AF
  028C3 2AA040          LD      HL,(40A0H)      ;HL -> start of string space
  028C6 EB              EX      DE,HL           ;DE = HL
  028C7 2AD640          LD      HL,(40D6H)      ;HL -> first free byte in
                                                ;string space (the string space
                                                ;is filled top down!)
  028CA 2F              CPL                     ;A = -A
  028CB 4F              LD      C,A             ;BC = number of required bytes
  028CC 06FF            LD      B,0FFH
  028CE 09              ADD     HL,BC           ;HL = HL + (-number of required
                                                ;bytes) = start address for
                                                ;the new string
  028CF 23              INC     HL              ;+ 1 (HL already pointed to a
                                                ;free byte)
  028D0 DF              RST     18H             ;Still inside string space ?
  028D1 3807            JR      C,28DAH         ;No: sort string space and try
                                                ;    again at 1st run
                                                ;    ?OS Error at 2nd run
                                                ;Yes:
  028D3 22D640          LD      (40D6H),HL      ;Store pointer
  028D6 23              INC     HL              ;Pointer + 1
  028D7 EB              EX      DE,HL           ;DE -> free space
  028D8 F1              POP     AF              ;Restore AF
  028D9 C9              RET


; Sort string space (garbage collection)
; Check if there are string in the string space, that belong to already
; cleared variables. If so, remove this garbage and move the used string
; space accordingly.

  028DA F1              POP     AF              ;Restore flag
                                                ;Already the 2nd run ?
  028DB 1E1A            LD      E,1AH           ;E = error code for ?OS Error
  028DD CAA219          JP      Z,19A2H         ;Yes: issue error

  028E0 BF              CP      A               ;Z-flag = 1 (mark 2nd run)
  028E1 F5              PUSH    AF              ;Save flag
  028E2 01C128          LD      BC,28C1H        ;Set new RET address
  028E5 C5              PUSH    BC              ;to 28C1H
  028E6 2AB140          LD      HL,(40B1H)      ;HL -> End of string space


; Search next highest string in string space and sort
; (highest string is the string with the highest start address)
; HL = address of string last inserted and sorted in string space
; (= address of last string in string space at last run)

  028E9 22D640          LD      (40D6H),HL      ;Store address of last string
                                                ;sorted
  028EC 210000          LD      HL,0000H        ;HL = 0000H (default address)
  028EF E5              PUSH    HL              ;Save it on stack
  028F0 2AA040          LD      HL,(40A0H)      ;HL -> start of string space
  028F3 E5              PUSH    HL              ;Save pointer on stack
```

```
; Test on highest string starts at the string space start address and ends
; on the last inserted and sorted string.
; The highest string in this range is then sorted and the routine is
; repeated until no highest string is found (all strings are sorted)
; Looks like bubblesort is used!

  028F4 21B540          LD      HL,40B5H         ;HL -> start of string table
                                                 ;= vector address of the first
                                                 ;string in the string table

; Test all string variables
; First the entries in the string table, then the string (single) variables
; and finally the string array variables are tested to see if they are
; located in string space

  028F7 EB              EX      DE,HL

  028F8 2AB340          LD      HL,(40B3H)       ;DE -> next free location in
                                                 ;string space
  028FB EB              EX      DE,HL
  028FC DF              RST     18H              ;End of table reached?
  028FD 01F728          LD      BC,28F7H         ;BC = new RET address
  02900 C24A29          JP      NZ,294AH         ;No : continue at 29A4H

  02903 2AF940          LD      HL,(40F9H)       ;HL -> start of BASIC var space
  02906 EB              EX      DE,HL
  02907 2AFB40          LD      HL,(40FBH)
  0290A EB              EX      DE,HL            ;DE -> start of array variables
  0290B DF              RST     18H              ;End of single variables
                                                 ;storage space reached?
  0290C 2813            JR      Z,2921H          ;Yes: continue at 2921H

  0290E 7E              LD      A,(HL)           ;A = type code
  0290F 23              INC     HL               ;Pointer + 1 (type code)
  02910 23              INC     HL               ;Pointer + 2 (name)
  02911 23              INC     HL
  02912 FE03            CP      03H              ;String variable found ?
  02914 2004            JR      NZ,291AH         ;No: test next variable
                                                 ;Yes:
  02916 CD4B29          CALL    294BH            ;Test address
  02919 AF              XOR     A                ;A = 00H
  0291A 5F              LD      E,A              ;DE = offset to next variable
  0291B 1600            LD      D,00H
  0291D 19              ADD     HL,DE            ;Add offset:
                                                 ;HL -> vector of next variable
  0291E 18E6            JR      2906H            ;Test next variable

; End of single variable space reached.
; Test array variables.

  02920 C1              POP     BC               ;Correct stack
  02921 EB              EX      DE,HL
  02922 2AFD40          LD      HL,(40FDH)       ;DE -> end of array variable
  02925 EB              EX      DE,HL            ;space
  02926 DF              RST     18H              ;End reached ?
  02927 CA6B29          JP      Z,296BH          ;Yes: insert highest string
                                                 ;and sort
                                                 ;No:
  0292A 7E              LD      A,(HL)           ;A = type code of array element
  0292B 23              INC     HL               ;Pointer + 1
  0292C CDC209          CALL    09C2H            ;BCDE = (HL): DE = name,
                                                 ;BC = length of array
  0292F E5              PUSH    HL               ;Save pointer
  02930 09              ADD     HL,BC            ;HL = pointer on next array
  02931 FE03            CP      03H              ;String array found ?
  02933 20EB            JR      NZ,2920H         ;No: test next array
                                                 ;Yes:
  02935 22D840          LD      (40D8H),HL       ;Save pointer
  02938 E1              POP     HL               ;HL -> number of dimensions
  02939 4E              LD      C,(HL)           ;BC = number of dimensions
  0293A 0600            LD      B,00H
```

```
  0293C 09              ADD     HL,BC           ;Add twice (every dimension
  0293D 09              ADD     HL,BC           ;is stored in 2 bytes)
                                                ;HL now contains the vector
                                                ;address of the first array
                                                ;element - 1
  0293E 23              INC     HL              ;HL + 1: vector
  0293F EB              EX      DE,HL           ;DE -> next array (= end address
  02940 2AD840          LD      HL,(40D8H)      ;of current array, see 2935H)
  02943 EB              EX      DE,HL
  02944 DF              RST     18H             ;End of array reached ?
  02945 28DA            JR      Z,2921H         ;Yes: test next array
                                                ;No:
  02947 013F29          LD      BC,293FH        ;Set new RET address
  0294A C5              PUSH    BC              ;to 293FH


; Check string address of found string variable
; Two criteria exist:
; 1. Is the string in string space and not reused again ?
;    (Is the string address smaller than the address of the last string
;     sorted and inserted ?)
; 2. Is the string located at a higher position in string space than the
;    string that is accepted to be the highest string in string space ?
;    (Is the string address bigger than the string address of the string
;     that is the highest accepted string ?)
;
; If both criteria are true, the address of the found string is stored on stack
; as the highest address.
;
; I: HL = vector address of string variable
; O: HL = vector address on the next variable

  0294B AF              XOR     A               ;A = 00H
  0294C B6              OR      (HL)            ;A = string length
  0294D 23              INC     HL              ;Pointer + 1
  0294E 5E              LD      E,(HL)          ;DE -> string
  0294F 23              INC     HL
  02950 56              LD      D,(HL)
  02951 23              INC     HL
                                                ;String length = 0 ?
  02952 C8              RET     Z               ;Yes: done, return
                                                ;No:
; 1st criterium

  02953 44              LD      B,H             ;BC -> vector of next variable
  02954 4D              LD      C,L
  02955 2AD640          LD      HL,(40D6H)      ;HL = address of last inserted
                                                ;variable
  02958 DF              RST     18H             ;Is the found string already
                                                ;re-inserted into the string
                                                ;space ?
                                                ;(Is the string address greater
                                                ;thean de address of the last
                                                ;inserted string ?)
  02959 60              LD      H,B             ;HL -> vector
  0295A 69              LD      L,C
  0295B D8              RET     C               ;Yes: done, return
                                                ;No:
; 2nd criterium

  0295C E1              POP     HL              ;HL = RET address
  0295D E3              EX      (SP),HL         ;Save RET address
                                                ;HL = address of last inserted
                                                ;highest string (see 2968H)
  0295E DF              RST     18H             ;Start address of string > HL ?
  0295F E3              EX      (SP),HL         ;Put HL and RET adrress
  02960 E5              PUSH    HL              ;back on the stack
  02961 60              LD      H,B             ;HL -> vector
  02962 69              LD      L,C
  02963 D0              RET     NC              ;No: done, return
                                                ;Yes:
```

```
; New highest string found
; Save string address and vector address on stack

  02964 C1              POP     BC              ;BC = RET address
  02965 F1              POP     AF              ;Remove start address of last
                                                ;inserted highest string
  02966 F1              POP     AF              ;and its vector address
                                                ;from stack
  02967 E5              PUSH    HL              ;Save current vector address
  02968 D5              PUSH    DE              ;and start address of string
                                                ;on stack
  02969 C5              PUSH    BC              ;Put RET address back on stack
  0296A C9              RET

; All stringvariables tested
; Sort and insert new highest string behind last sorted and inserted string
; and clear all garbage in between.

  0296B D1              POP     DE              ;Take address of highest string
  0296C E1              POP     HL              ;and the vector address
  0296D 7D              LD      A,L             ;Address changes since 28ECH ?
  0296E B4              OR      H               ;(new highest string found ?)
  0296F C8              RET     Z               ;No: done, return

; A new highest string has been found

  02970 2B              DEC     HL              ;BC = address of new highest
  02971 46              LD      B,(HL)          ;string
  02972 2B              DEC     HL              ;( = DE because the address
  02973 4E              LD      C,(HL)          ;belongs to DE )
  02974 E5              PUSH    HL              ;Save address + 1
  02975 2B              DEC     HL              ;HL - 1 = vector address
                                                ;of last tested string
  02976 6E              LD      L,(HL)
  02977 2600            LD      H,00H           ;HL = string length
  02979 09              ADD     HL,BC           ;HL = length + string address
  0297A 50              LD      D,B             ;DE = string address
  0297B 59              LD      E,C
  0297C 2B              DEC     HL              ;HL = end address of string
  0297D 44              LD      B,H             ;BC = HL
  0297E 4D              LD      C,L
  0297F 2AD640          LD      HL,(40D6H)      ;HL = address of last inserted
                                                ;string
  02982 CD5819          CALL    1958H           ;Insert new string by attaching
                                                ;the new string to the last
                                                ;inserted string
  02985 E1              POP     HL              ;Restore vector address + 1
  02986 71              LD      (HL),C          ;Use new start address
  02987 23              INC     HL
  02988 70              LD      (HL),B
  02989 69              LD      L,C             ;HL = start address of inserted
  0298A 60              LD      H,B             ;string
  0298B 2B              DEC     HL              ;HL - 1 = address of last
                                                ;inserted string
  0298C C3E928          JP      28E9H           ;Search next highest string


; String addition
;
; I: HL = PTP on '+'
;    BC = priority code
;    X  = 1st string
; O: X  = new string

  0298F C5              PUSH    BC              ;Save priority code
  02990 E5              PUSH    HL              ;Save PTP
  02991 2A2141          LD      HL,(4121H)      ;HL -> vector of 1st string
  02994 E3              EX      (SP),HL         ;Save vector adrress
                                                ;and restore PTP
  02995 CD9F24          CALL    249FH           ;X = 2nd string
  02998 E3              EX      (SP),HL         ;Save PTP and restore vector
```

```
                                                 ;of 1st string
  02999 CDF40A          CALL    0AF4H            ;?TM Error if X is not STR type
  0299C 7E              LD      A,(HL)           ;A = length of 1st string
  0299D E5              PUSH    HL               ;Save vector address of
                                                 ;1st string
  0299E 2A2141          LD      HL,(4121H)       ;HL -> vector of 2nd string
  029A1 E5              PUSH    HL               ;Save vector address of
                                                 ;2nd string
  029A2 86              ADD     A,(HL)           ;A = total length of new
                                                 ;string
  029A3 1E1C            LD      E,1CH            ;E = error code for ?LS Error
                                                 ;Overflow ? (new string too
                                                 ;long)
  029A5 DAA219          JP      C,19A2H          ;Yes: ?LS Error
                                                 ;No:
  029A8 CD5728          CALL    2857H            ;Reserve A bytes of free space
                                                 ;in string space. HL -> vector
                                                 ;new string (space)
  029AB D1              POP     DE               ;Restore vector address of
                                                 ;2nd string
  029AC CDDE29          CALL    29DEH            ;Delete 2nd string from string
                                                 ;table and string space ?
  029AF E3              EX      (SP),HL          ;Save vector address of 2nd
                                                 ;string and restore vector
                                                 ;address of 1st string
  029B0 CDDD29          CALL    29DDH            ;Delete 1nd string from string
                                                 ;table and string space ?
  029B3 E5              PUSH    HL               ;Save vector address of
                                                 ;1st string
  029B4 2AD440          LD      HL,(40D4H)       ;HL = address of free space for
                                                 ;new string
  029B7 EB              EX      DE,HL            ;DE = HL
  029B8 CDC629          CALL    29C6H            ;Copy 1st string into new space
  029BB CDC629          CALL    29C6H            ;Copy 2nd string behind 1st
                                                 ;string
  029BE 214923          LD      HL,2349H         ;Jump back to decoding of
                                                 ;expression ar 2349H
  029C1 E3              EX      (SP),HL          ;New RET address on stack
  029C2 E5              PUSH    HL               ;Save old RET address on stack
  029C3 C38428          JP      2884H            ;Insert new string into
                                                 ;string table

; Copy string to (DE)
;
; I: (SP - 2) -> string vector
;          DE =  new string address

  029C6 E1              POP     HL               ;RET address to HL
  029C7 E3              EX      (SP),HL          ;Get vector address and put
                                                 ;RET address back on stack
  029C8 7E              LD      A,(HL)           ;A = length
  029C9 23              INC     HL
  029CA 4E              LD      C,(HL)           ;BC = string address
  029CB 23              INC     HL
  029CC 46              LD      B,(HL)
  029CD 6F              LD      L,A              ;L = counter
  029CE 2C              INC     L                ;Adjust counter

  029CF 2D              DEC     L                ;Counter - 1
                                                 ;Counter = 0 ?
  029D0 C8              RET     Z                ;Yes: done, return
                                                 ;No:
  029D1 0A              LD      A,(BC)           ;Copy byte from (BC)
  029D2 12              LD      (DE),A           ;to (DE)
  029D3 03              INC     BC               ;Source pointer + 1
  029D4 13              INC     DE               ;Destination pointer + 1
  029D5 18F8            JR      29CFH            ;Next byte


; Remove string in X from string table and string space ?
;
```

```
; When using string-functions all string constants and intermediate results
; are basically intermediate in the string table and string space.
; Since these strings are only used by the function, they can be removed
; after completion of the function
;
; I: X   = string
; O: HL -> string vector

  029D7 CDF40A          CALL    0AF4H           ;Test if X in STR format
  029DA 2A2141          LD      HL,(4121H)      ;HL -> string vector

; Remove from string table and string space ?

  029DD EB              EX      DE,HL           ;DE -> string vector
  029DE CDF529          CALL    29F5H           ;If the vector address points
                                                ;to the last entry in the
                                                ;string table, it will be
                                                ;deleted (in this case it
                                                ;is an intermediate result or
                                                ;a string constant)
  029E1 EB              EX      DE,HL           ;HL -> string vector
                                                ;Last entry deleted ?
  029E2 C0              RET     NZ              ;No: done, return

  029E3 D5              PUSH    DE              ;Save vector address
  029E4 50              LD      D,B             ;BC = string address
  029E5 59              LD      E,C
  029E6 1B              DEC     DE              ;DE - 1
  029E7 4E              LD      C,(HL)          ;C = string lenght
  029E8 2AD640          LD      HL,(40D6H)      ;HL = address of last string
                                                ;in string space - 1
  029EB DF              RST     18H             ;Is the string the last one
                                                ;in string space ? (this means
                                                ;it is a string constant !)
  029EC 2005            JR      NZ,29F3H        ;No: done
                                                ;Yes:
  029EE 47              LD      B,A             ;B = 0, BC = string length
  029EF 09              ADD     HL,BC           ;Add string length to HL
                                                ;HL now points on the new last
                                                ;string in string space
  029F0 22D640          LD      (40D6H),HL      ;Store address
                                                ;(the string constant is now
                                                ;removbed from string space)
  029F3 E1              POP     HL              ;Restore vector address
  029F4 C9              RET


; Does DE point to the last entry of the string table ?
; If it does, then the last entry is deleted.
;
; I: DE -> string vector
; O: BC =  string address of last entry in string table
;    DE -> string vector
;    HL =  DE
;    Z-flag = 1: the entry was deleted

  029F5 2AB340          LD      HL,(40B3H)      ;HL -> next free entry of
                                                ;string table
  029F8 2B              DEC     HL
  029F9 46              LD      B,(HL)          ;BC = string address
  029FA 2B              DEC     HL
  029FB 4E              LD      C,(HL)
  029FC 2B              DEC     HL
  029FD DF              RST     18H             ;Does DE point to this entry ?
  029FE C0              RET     NZ              ;No: done, return
                                                ;Yes:
  029FF 22B340          LD      (40B3H),HL      ;Save new address
                                                ;(overwrite old entry)
  02A02 C9              RET
```

```
; X = LEN ( X )
; -------------

  02A03 01F827          LD      BC,27F8H        ;Set new return address
  02A06 C5              PUSH    BC              ;to 27F8H

  02A07 CDD729          CALL    29D7H           ;Test X has STR format
                                                ;Remove string from string
                                                ;table and string space ?
                                                ;HL -> vector of string arg.
  02A0A AF              XOR     A               ;A = 00H
  02A0B 57              LD      D,A             ;D = 00H
  02A0C 7E              LD      A,(HL)          ;A = string length
  02A0D B7              OR      A               ;Set flags
  02A0E C9              RET                     ;Return to 27F8H: write A to X
                                                ;as INT

; X = ASC (  )
; ------------

  02A0F 01F827          LD      BC,27F8H        ;Set new return address
  02A12 C5              PUSH    BC              ;to 27F8H

  02A13 CD072A          CALL    2A07H           ;Get string length and vector
                                                ;address
                                                ;Length zero ?
  02A16 CA4A1E          JP      Z,1E4AH         ;Yes: ?FC Error
                                                ;HL -> string variable
  02A19 23              INC     HL
  02A1A 5E              LD      E,(HL)          ;DE -> string
  02A1B 23              INC     HL
  02A1C 56              LD      D,(HL)

  02A1D 1A              LD      A,(DE)          ;A = first string character
  02A1E C9              RET                     ;Return to 27F8H: write A to X
                                                ;as INT

; X = CHR$ (  )
; -------------

  02A1F 3E01            LD      A,01H           ;A = length of resulting string
  02A21 CD5728          CALL    2857H           ;Create room in string space
  02A24 CD1F2B          CALL    2B1FH           ;Argument to DE
  02A27 2AD440          LD      HL,(40D4H)      ;HL -> string
  02A2A 73              LD      (HL),E          ;Write ASCII value into string
  02A2B C1              POP     BC              ;Remove RET address
  02A2C C38428          JP      2884H           ;Put string in string space


; X = STRING$
; -----------

  02A2F D7              RST     10H             ;Adjust PTP
  02A30 CF              RST     08H             ;Next character must be
  02A31 28              DEFB    '('             ;a '('
  02A32 CD1C2B          CALL    2B1CH           ;Get and save string length
  02A35 D5              PUSH    DE
  02A36 CF              RST     08H             ;Next character must be
  02A37 2C              DEFB    ','             ;a comma
  02A38 CD3723          CALL    2337H           ;Put argument into X
  02A3B CF              RST     08H             ;Next character must be
  02A3C 29              DEFB    ')'             ;a ')'
  02A3D E3              EX      (SP),HL         ;Save PTP, restore length
  02A3E E5              PUSH    HL              ;Save length
  02A3F E7              RST     20H             ;TSTTYP: STR argument ?
  02A40 2805            JR      Z,2A47H         ;Yes: continue at 2A47H
                                                ;No:
  02A42 CD1F2B          CALL    2B1FH           ;Is the number in range
                                                ;from 0 to 255 ? No: ?FC Error
  02A45 1803            JR      2A4AH           ;Continue at 2A4AH (A = number)
```

```
  02A47 CD132A          CALL    2A13H           ;A = ASCII value of first
                                                ;character of string argument
  02A4A D1              POP     DE              ;Restore length
  02A4B F5              PUSH    AF              ;Dummy push because of jump
                                                ;to 2A2BH after ending
  02A4C F5              PUSH    AF              ;Save character
  02A4D 7B              LD      A,E             ;A = length
  02A4E CD5728          CALL    2857H           ;Reserve space
  02A51 5F              LD      E,A             ;E = counter
  02A52 F1              POP     AF              ;Restore character
  02A53 1C              INC     E               ;Counter = 0?
  02A54 1D              DEC     E
  02A55 28D4            JR      Z,2A2BH         ;Yes: use string
                                                ;No:
  02A57 2AD440          LD      HL,(40D4H)      ;HL -> free space
  02A5A 77              LD      (HL),A          ;Store character
  02A5B 23              INC     HL              ;Pointer + 1
  02A5C 1D              DEC     E               ;Counter - 1, counter = 0 ?
  02A5D 20FB            JR      NZ,2A5AH        ;No: next character
                                                ;Yes:
  02A5F 18CA            JR      2A2BH           ;Use string


; LEFT$
; -----
;
; I: (SP - 2) -> vector of the string argument
;    (SP)     =  number

  02A61 CDDF2A          CALL    2ADFH           ;')'?
                                                ;B = number = new string length
  02A64 AF              XOR     A               ;Set starting point on 0

; Entry for RIGHT$
;
; I: A = starting point of new string
;    B = length of new string

  02A65 E3              EX      (SP),HL         ;Save PTP, HL -> vector
  02A66 4F              LD      C,A             ;C = A = starting point of
                                                ;new string - 1
  02A67 3EE5            LD      A,0E5H          ;--

; Entry for MID$
;
; I: B = length of new string
;    C = starting point of new string inside string argument - 1

* 02A69   E5            PUSH    HL              ;Save vector address

; Get substring from string argument (in X) and store in X
;
; I: X = string argument
;    B = length of new string
;    C = starting point of new string inside string argument - 1
; O: X = requested substring

  02A69 E5              PUSH    HL              ;Save vector address
  02A6A 7E              LD      A,(HL)          ;A = length of argument string
  02A6B B8              CP      B               ;Substring longer than
                                                ;argument string ?
  02A6C 3802            JR      C,2A70H         ;Yes: set C to 0 and use the
                                                ;complete argument string
                                                ;No:
  02A6E 78              LD      A,B             ;A = required length
  02A6F 110E00          LD      DE,000EH        ;--

* 02A70   0E00          LD      C,00H           ;Starting point = 0
  02A72 C5              PUSH    BC              ;Save BC
  02A73 CDBF28          CALL    28BFH           ;Free up A bytes of space
                                                ;DE -> free space
```

```
  02A76 C1               POP     BC              ;Restore BC
  02A77 E1               POP     HL              ;Resore vector address
  02A78 E5               PUSH    HL              ;and save it again
  02A79 23               INC     HL              ;Vector address + 1
  02A7A 46               LD      B,(HL)          ;HL -> argument string
  02A7B 23               INC     HL
  02A7C 66               LD      H,(HL)
  02A7D 68               LD      L,B
  02A7E 0600             LD      B,00H           ;BC = offset to new starting
                                                 ;point
  02A80 09               ADD     HL,BC           ;HL -> start of substring
  02A81 44               LD      B,H             ;BC -> start of substring
  02A82 4D               LD      C,L
  02A83 CD5A28           CALL    285AH           ;Store length and address of
                                                 ;new string in string table
  02A86 6F               LD      L,A             ;L = length of substring
  02A87 CDCE29           CALL    29CEH           ;Copy characters of argument
                                                 ;string to (DE)
  02A8A D1               POP     DE              ;Restore vector address of
                                                 ;argument string
  02A8B CDDE29           CALL    29DEH           ;Delete argument string from
                                                 ;string space and string table?
  02A8E C38428           JP      2884H           ;Use new string


; RIGHT$
; ------
;
; I: (SP - 2) -> vector of the string argument
;    (SP)     =  number (start of new string)

  02A91 CDDF2A           CALL    2ADFH           ;')'? B = number
  02A94 D1               POP     DE              ;DE = vector address
  02A95 D5               PUSH    DE              ;Back into stack
  02A96 1A               LD      A,(DE)          ;A = argument string length
  02A97 90               SUB     B               ;A = length - number =
                                                 ;starting point of new string
  02A98 18CB             JR      2A65H           ;Continue at 2A65H


; MID$
; ----
; (on the right hand side of the equal sign
;
; I: (SP - 2) -> vector of the string argument
;    (SP)     =  number (start of new string)

  02A9A EB               EX      DE,HL           ;HL = PTP
  02A9B 7E               LD      A,(HL)          ;A = next character
  02A9C CDE22A           CALL    2AE2H           ;B = starting point
  02A9F 04               INC     B               ;Starting point = 0 ?
  02AA0 05               DEC     B
  02AA1 CA4A1E           JP      Z,1E4AH         ;Yes: ?FC Error
                                                 ;No:
  02AA4 C5               PUSH    BC              ;Save starting point
  02AA5 1EFF             LD      E,0FFH          ;E = default length (in case
                                                 ;of missing length indication
                                                 ;the total string length
                                                 ;from starting point onwards
                                                 ;is used)
  02AA7 FE29             CP      ')'             ;Next character = ')' ?
  02AA9 2805             JR      Z,2AB0H         ;Yes: E = length
                                                 ;No:
  02AAB CF               RST     08H             ;Next character must be
  02AAC 2C               DEFB    ','             ;a comma
  02AAD CD1C2B           CALL    2B1CH           ;Get length and store in E
  02AB0 CF               RST     08H             ;Next character must be
  02AB1 29               DEFB    ')'             ;a ')'
  02AB2 F1               POP     AF              ;A = starting point
  02AB3 E3               EX      (SP),HL         ;Save PTP, restore vector addr.
  02AB4 01692A           LD      BC,2A69H        ;Set new RET address
```

```
  02AB7 C5              PUSH    BC              ;to 2A69H
  02AB8 3D              DEC     A               ;A = starting point - 1
  02AB9 BE              CP      (HL)            ;Compare with length of
                                                ;argument string
  02ABA 0600            LD      B,00H           ;Length of new string = 0
  02ABC D0              RET     NC              ;Create string with length = 0
                                                ;in case starting point is
                                                ;larger than string length
  02ABD 4F              LD      C,A             ;C = starting point - 1
  02ABE 7E              LD      A,(HL)          ;A = length of arg. string
  02ABF 91              SUB     C               ;- statring point =
                                                ;remaining string length
  02AC0 BB              CP      E               ;Requested length >
                                                ;remaining length
  02AC1 47              LD      B,A             ;B = remaining length
  02AC2 D8              RET     C               ;Yes: B = length
                                                ;No:
  02AC3 43              LD      B,E             ;B = requested length
  02AC4 C9              RET                     ;Use this


; X = VAL (  )
; ------------

  02AC5 CD072A          CALL    2A07H           ;Get vector address and length
                                                ;of argument. D = 00H
                                                ;String length zero ?
  02AC8 CAF827          JP      Z,27F8H         ;Yes: result = 0.
                                                ;No:
  02ACB 5F              LD      E,A             ;E = length
  02ACC 23              INC     HL              ;Vector address + 1
  02ACD 7E              LD      A,(HL)          ;HL -> string
  02ACE 23              INC     HL
  02ACF 66              LD      H,(HL)
  02AD0 6F              LD      L,A
  02AD1 E5              PUSH    HL              ;Save pointer
  02AD2 19              ADD     HL,DE           ;Calculate string end address
  02AD3 46              LD      B,(HL)          ;Get next character
  02AD4 72              LD      (HL),D          ;and replace it by 00H
  02AD5 E3              EX      (SP),HL         ;Save end address, restore
                                                ;start address
  02AD6 C5              PUSH    BC              ;Save character
  02AD7 7E              LD      A,(HL)          ;A = 1st character
  02AD8 CD650E          CALL    0E65H           ;Decode string, X = number
  02ADB C1              POP     BC              ;Restore characater
  02ADC E1              POP     HL              ;Restore end address
  02ADD 70              LD      (HL),B          ;Insert character
  02ADE C9              RET


; SUB for LEFT$, RIGHT$ and MID$
; Test for closing bracket and get first number from stack

  02ADF EB              EX      DE,HL           ;HL = PTP
  02AE0 CF              RST     08H             ;Next character must be
  02AE1 29              DEFB    ')'             ;a ')'
  02AE2 C1              POP     BC              ;BC = RET address
  02AE3 D1              POP     DE              ;DE = number
  02AE4 C5              PUSH    BC              ;Put RET address on stack
  02AE5 43              LD      B,E             ;B = number
  02AE6 C9              RET


; Token found not in the range from 80H to BBH (see 1D67H)
; (not a command)

  02AE7 FE7A            CP      7AH             ;Is it MID$ ?
                                                ;(left of '=' sign !)
  02AE9 C29719          JP      NZ,1997H        ;No: ?SN Error

  02AEC C3D941          JP      41D9H           ;Yes: DOS
```

```
; IN statement
; ------------

  02AEF CD1F2B          CALL    2B1FH           ;A = port address
  02AF2 329440          LD      (4094H),A       ;Save it
  02AF5 CD9340          CALL    4093H           ;Perform IN
  02AF8 C3F827          JP      27F8H           ;Write A to X as INT


; OUT statement
; -------------

  02AFB CD0E2B          CALL    2B0EH           ;Get port address and value
  02AFE C39640          JP      4096H           ;Perform OUT


; Decode argument at (HL) and store in X as an INT
;
; I: HL = PTP
; O: DE = number
;    A = MSB of number
;    Z-flag = 1 if number < 256 (MSB = 0)

  02B01 D7              RST     10H             ;Get next non-space character
  02B02 CD3723          CALL    2337H           ;Decode argument
  02B05 E5              PUSH    HL              ;Save PTP
  02B06 CD7F0A          CALL    0A7FH           ;HL = X = CINT ( X )
  02B09 EB              EX      DE,HL           ;DE = INT number
  02B0A E1              POP     HL              ;Restore PTP
  02B0B 7A              LD      A,D             ;A = MSB
  02B0C B7              OR      A               ;MSB = 0 ?
  02B0D C9              RET


; SUB for OUT
; Get port address and prepare RAM at 4096H

  02B0E CD1C2B          CALL    2B1CH           ;Get port address
  02B11 329440          LD      (4094H),A       ;Store for IN
  02B14 329740          LD      (4097H),A       ;and OUT
  02B17 CF              RST     08H             ;Next character must be
  02B18 2C              DEFB    ','             ;a comma
  02B19 1801            JR      2B1CH           ;Get value and return


; Decode argument at (HL) and store in X as INT
; ?FC Error if the result is not in the range from 0 to 255
;
; I: HL = PTP
; O: A = number
;    DE = number

  02B1B D7              RST     10H             ;Get next non-space character
  02B1C CD3723          CALL    2337H           ;Decode argument
  02B1F CD052B          CALL    2B05H           ;In range [0..255] ?
  02B22 C24A1E          JP      NZ,1E4AH        ;No: ?FC Error

  02B25 2B              DEC     HL              ;PTP - 1
  02B26 D7              RST     10H             ;And increment again
  02B27 7B              LD      A,E             ;A = number
  02B28 C9              RET


; LLIST statement
; ---------------

  02B29 3E01            LD      A,01H           ;Set output flag to
  02B2B 329C40          LD      (409CH),A       ;printer output
```

```
; LIST statement
; --------------

  02B2E C1              POP     BC              ;Remove RET address
  02B2F CD101B          CALL    1B10H           ;Get line number
  02B32 C5              PUSH    BC              ;LP on first line
  02B33 21FFFF          LD      HL,0FFFFH       ;Set current LN to 65535
  02B36 22A240          LD      (40A2H),HL
  02B39 E1              POP     HL              ;HL = LP on line
  02B3A D1              POP     DE              ;DE = LN of last line
  02B3B 4E              LD      C,(HL)          ;BC = LP on next line
  02B3C 23              INC     HL
  02B3D 46              LD      B,(HL)
  02B3E 23              INC     HL
  02B3F 78              LD      A,B             ;End of program reached ?
  02B40 B1              OR      C
  02B41 CA191A          JP      Z,1A19H         ;Yes: done, continue at 1A19H

  02B44 CDDF41          CALL    41DFH           ;DOS
  02B47 CD9B1D          CALL    1D9BH           ;<SHIFT>+<@> or
                                                ;<BREAK> pressed ?
  02B4A C5              PUSH    BC              ;Save LP
  02B4B 4E              LD      C,(HL)          ;BC = line number
  02B4C 23              INC     HL              ;of current line
  02B4D 46              LD      B,(HL)
  02B4E 23              INC     HL
  02B4F C5              PUSH    BC              ;Save line number
  02B50 E3              EX      (SP),HL         ;Save pointer, restore LN
  02B51 EB              EX      DE,HL           ;DE = current line number
                                                ;HL = last line number
  02B52 DF              RST     18H             ;Current line number > last
                                                ;line number ?
  02B53 C1              POP     BC              ;Pointer on current line
                                                ;back in BC
  02B54 DA181A          JP      C,1A18H         ;Yes: done, continue at 1A18H

  02B57 E3              EX      (SP),HL         ;Save last line number
                                                ;HL = LP on next line
  02B58 E5              PUSH    HL              ;Save LP on next line
  02B59 C5              PUSH    BC              ;Save LP on current line
  02B5A EB              EX      DE,HL           ;HL = current line number
  02B5B 22EC40          LD      (40ECH),HL      ;Save as '.'-line
  02B5E CDAF0F          CALL    0FAFH           ;Print HL as decimal number
  02B61 3E20            LD      A,20H           ;A = ' '
  02B63 E1              POP     HL              ;Restore pointer on current
                                                ;line
  02B64 CD2A03          CALL    032AH           ;Print space
  02B67 CD7E2B          CALL    2B7EH           ;Decode line from (HL) onwards
                                                ;and store in line buffer
  02B6A 2AA740          LD      HL,(40A7H)      ;HL -> start of line buffer
  02B6D CD752B          CALL    2B75H           ;Print line
  02B70 CDFE20          CALL    20FEH           ;Start new line
  02B73 18BE            JR      2B33H           ;Process next line


; SUB for LIST
; Print text from (HL) onwards, 00H = end of text
;
; I: HL -> text to be printed
; O: HL -> end of text (00H)

  02B75 7E              LD      A,(HL)          ;A = next character
  02B76 B7              OR      A               ;End of text ?
  02B77 C8              RET     Z               ;Yes: done, return

  02B78 CD2A03          CALL    032AH           ;Print it
  02B7B 23              INC     HL              ;Pointer + 1
  02B7C 18F7            JR      2B75H           ;Next character
```

```
; SUB for LIST and EDIT
; Decode line from (HL) onwards and store result in line buffer
;
; I: HL -> program text


  02B7E E5              PUSH    HL              ;Save pointer
  02B7F 2AA740          LD      HL,(40A7H)      ;HL -> start of line buffer
  02B82 44              LD      B,H             ;BC -> start of line buffer
  02B83 4D              LD      C,L
  02B84 E1              POP     HL              ;Restore pointer
  02B85 16FF            LD      D,0FFH          ;D = counter (maximum length =
                                                ;255 characters)
  02B87 1803            JR      2B8CH           ;Continue at 2B8CH

; Decode next character

  02B89 03              INC     BC              ;Pointer + 1
  02B8A 15              DEC     D               ;Counter - 1
                                                ;Maximum length reached ?
  02B8B C8              RET     Z               ;Yes: done, return

; Decode line from (HL) onwards and store at (BC)

  02B8C 7E              LD      A,(HL)          ;A = next character
  02B8D B7              OR      A               ;End of line reached ?
  02B8E 23              INC     HL              ;Pointer + 1
  02B8F 02              LD      (BC),A          ;First store character
  02B90 C8              RET     Z               ;Yes: done, return
                                                ;Token found ?
  02B91 F2D23F          JP      P,3FD2H         ;No: continue at 3FD2H

  02B94 FEFB            CP      0FBH            ;"'" token found ? (REM)
  02B96 2008            JR      NZ,2BA0H        ;No: continue at 2BA0H

  02B98 0B              DEC     BC              ;Yes: delete last 4 characters
  02B99 0B              DEC     BC              ;in line buffer because the
  02B9A 0B              DEC     BC              ;apostroph is stored as
  02B9B 0B              DEC     BC              ;'::REM'
  02B9C 14              INC     D               ;Counter + 1
  02B9D 14              INC     D
  02B9E 14              INC     D
  02B9F 14              INC     D
  02BA0 FE95            CP      95H             ;'ELSE' token found ?
  02BA2 CC240B          CALL    Z,0B24H         ;Yes: buffer pointer - 1
                                                ;('ELSE' is stored as ':ELSE')
  02BA5 D67F            SUB     7FH             ;A = token value - 7FH
  02BA7 E5              PUSH    HL              ;Save pointer
  02BA8 5F              LD      E,A             ;E = token value
  02BA9 CDAD39          CALL    39ADH           ;Colour-Token found ?
                                                ;HL -> start of corresponding
                                                ;keyword table
  02BAC 7E              LD      A,(HL)          ;A = next character
  02BAD B7              OR      A               ;Next keyword reached ?
  02BAE 23              INC     HL              ;Pointer + 1
  02BAF F2AC2B          JP      P,2BACH         ;No: increment HL to next
                                                ;keyword
  02BB2 1D              DEC     E               ;Token counter - 1
  02BB3 20F7            JR      NZ,2BACH        ;Increment pointer further
                                                ;until right token is reached
  02BB5 E67F            AND     7FH             ;Clear bit 7
  02BB7 02              LD      (BC),A          ;Store character
  02BB8 03              INC     BC              ;Buffer pointer + 1
  02BB9 15              DEC     D               ;Counter - 1
                                                ;Buffer full ?
  02BBA CAD828          JP      Z,28D8H         ;Yes: done, continue at 28D8H

  02BBD 7E              LD      A,(HL)          ;A = next character
  02BBE 23              INC     HL              ;Table pointer + 1
  02BBF B7              OR      A               ;Next keyword reached ?
  02BC0 F2B72B          JP      P,2BB7H         ;No: store character in buffer
```

```
  02BC3 E1              POP     HL              ;Yes: restore line pointer
  02BC4 18C6            JR      2B8CH           ;Process next character


; DELETE statement
; ----------------

  02BC6 CD101B          CALL    1B10H           ;Get start and end LN
  02BC9 D1              POP     DE              ;DE = LN of end line
  02BCA C5              PUSH    BC              ;Save pointer on start line
  02BCB C5              PUSH    BC              ;twice
  02BCC CD2C1B          CALL    1B2CH           ;Search end line
                                                ;Does it exist?
  02BCF 3005            JR      NC,2BD6H        ;No: ?FC Error
                                                ;Yes:
  02BD1 54              LD      D,H             ;DE = end line pointer
  02BD2 5D              LD      E,L
  02BD3 E3              EX      (SP),HL         ;Save pointer to end line
                                                ;HL = pointer to start line
  02BD4 E5              PUSH    HL              ;Save it
  02BD5 DF              RST     18H             ;Compare both pointer
                                                ;End >= start ?
  02BD6 D24A1E          JP      NC,1E4AH        ;No: ?FC Error
                                                ;Yes:
  02BD9 212919          LD      HL,1929H        ;HL -> 'READY'
  02BDC CDA728          CALL    28A7H           ;Print text
  02BDF C1              POP     BC              ;Restore pointer to start line
  02BE0 21E81A          LD      HL,1AE8H        ;Set new RET address to 1AE8H
  02BE3 E3              EX      (SP),HL         ;and restore end line pointer

; Delete line(s)
;
; I: BC -> start line
;    HL -> line following end line
;

  02BE4 EB              EX      DE,HL           ;DE -> line following end
  02BE5 2AF940          LD      HL,(40F9H)      ;HL -> end of program
  02BE8 1A              LD      A,(DE)          ;Copy next line
  02BE9 02              LD      (BC),A          ;onto line to be deleted
  02BEA 03              INC     BC              ;Pointer + 1
  02BEB 13              INC     DE              ;Pointer + 1
  02BEC DF              RST     18H             ;End of program reached ?
  02BED 20F9            JR      NZ,2BE8H        ;No: continue copying
                                                ;Yes:
  02BEF 60              LD      H,B             ;HL = new end address
  02BF0 69              LD      L,C             ;of program
  02BF1 22F940          LD      (40F9H),HL      ;Save it
  02BF4 C9              RET


; CSAVE statement
; ---------------

  02BF5 CD3723          CALL    2337H           ;Get filename
  02BF8 E5              PUSH    HL              ;Save PTP
  02BF9 CD132A          CALL    2A13H           ;DE -> filename
  02BFC F5              PUSH    AF              ;Save registers
  02BFD C5              PUSH    BC
  02BFE D5              PUSH    DE
  02BFF E5              PUSH    HL
  02C00 CD3F02          CALL    023FH           ;Write leader and sync
  02C03 E1              POP     HL              ;Restore registers
  02C04 D1              POP     DE
  02C05 C1              POP     BC
  02C06 F1              POP     AF
  02C07 1A              LD      A,(DE)          ;A = filename character
  02C08 CD1F02          CALL    021FH           ;Write to cassette
  02C0B 2AA440          LD      HL,(40A4H)      ;HL -> start of program
  02C0E EB              EX      DE,HL           ;DE = HL
  02C0F 2AF940          LD      HL,(40F9H)      ;HL -> end of program
```

```
  02C12 1A              LD      A,(DE)          ;A = byte from program text
  02C13 13              INC     DE              ;Program pointer + 1
  02C14 CD1F02          CALL    021FH           ;Write program byte to cassette
  02C17 DF              RST     18H             ;End of program reached ?
  02C18 20F8            JR      NZ,2C12H        ;No, write next byte

  02C1A 00              NOP                     ;--
  02C1B 00              NOP
  02C1C 00              NOP
  02C1D E1              POP     HL              ;Restore PTP
  02C1E C9              RET


; CLOAD statement
; ---------------

  02C1F 00              NOP                     ;--
  02C20 00              NOP
  02C21 00              NOP
  02C22 00              NOP
  02C23 00              NOP
  02C24 00              NOP
  02C25 00              NOP
  02C26 00              NOP
  02C27 AF              XOR     A               ;A = 00H
  02C28 012F23          LD      BC,232FH        ;--

; Entry for VERIFY

* 02C29   2F            CPL                     ;A <> 00H
* 02C2A     23          INC     HL              ;PTP + 1
  02C2B F5              PUSH    AF              ;Save flag
  02C2C 2B              DEC     HL              ;PTP - 1
  02C2D D7              RST     10H             ;Filename indicated ?
  02C2E 3E00            LD      A,00H           ;A = default filename
  02C30 2807            JR      Z,2C39H         ;No: continue at 2C39H
                                                ;Yes:
  02C32 CD3723          CALL    2337H           ;Put filename
  02C35 CD132A          CALL    2A13H           ;into A
  02C38 1A              LD      A,(DE)
  02C39 6F              LD      L,A             ;L = filename
  02C3A F1              POP     AF              ;Restore flag
  02C3B B7              OR      A               ;CLOAD ?
  02C3C 67              LD      H,A             ;H = flag
  02C3D 222141          LD      (4121H),HL      ;Save flag and filename
  02C40 CC4D1B          CALL    Z,1B4DH         ;Yes: NEW
  02C43 2A2141          LD      HL,(4121H)      ;Restore flag and filename
  02C46 EB              EX      DE,HL           ;D = flag, E = filename
  02C47 F5              PUSH    AF              ;Save registers
  02C48 C5              PUSH    BC
  02C49 D5              PUSH    DE
  02C4A E5              PUSH    HL
  02C4B CD4C02          CALL    024CH           ;Search for leader and sync
  02C4E E1              POP     HL              ;Restore registers
  02C4F D1              POP     DE
  02C50 C1              POP     BC
  02C51 F1              POP     AF
  02C52 CDED01          CALL    01EDH           ;Get filename char from tape
  02C55 1C              INC     E               ;Filename indicated ?
  02C56 1D              DEC     E
  02C57 2803            JR      Z,2C5CH         ;No: continue at 2C5CH

  02C59 BB              CP      E               ;Specified filename found ?
  02C5A 2037            JR      NZ,2C93H        ;No: search for next file

  02C5C 2AA440          LD      HL,(40A4H)      ;HL = program start address
  02C5F 0603            LD      B,03H           ;B = counter ( 3 times 00H
                                                ;indicates program end)
  02C61 CDED01          CALL    01EDH           ;Read 1 byte
  02C64 5F              LD      E,A             ;E = byte
```

```
  02C65 96             SUB    (HL)            ;Does memory also contain this
                                              ;byte ? Yes: A = 00H
  02C66 A2             AND    D               ;AND it with flag
                                              ;A = 00H if CLOAD or VERIFY
                                              ;is ok. A <> 00H if VERIFY is
                                              ;bad.
                                              ;Bad VERIFY ?
  02C67 2021           JR     NZ,2C8AH        ;Yes: continue at 2C8AH

  02C69 73             LD     (HL),E          ;Store byte
  02C6A CD6C19         CALL   196CH           ;Check free memory remaining.
  02C6D 7E             LD     A,(HL)          ;Byte read = 00H ?
  02C6E B7             OR     A
  02C6F 23             INC    HL              ;Pointer + 1
  02C70 20ED           JR     NZ,2C5FH        ;No: next byte
                                              ;Yes:
  02C72 CDE401         CALL   01E4H           ;Blink '*'
  02C75 10EA           DJNZ   2C61H           ;Counter - 1, read next byte

  02C77 22F940         LD     (40F9H),HL      ;Store new end of program
  02C7A 212919         LD     HL,1929H        ;HL -> 'READY'
  02C7D CDA728         CALL   28A7H           ;Print text
  02C80 00             NOP                    ;--
  02C81 00             NOP
  02C82 00             NOP
  02C83 2AA440         LD     HL,(40A4H)      ;Reset PTP to start of BASIC
                                              ;program
  02C86 E5             PUSH   HL              ;Save PTP
  02C87 C3E81A         JP     1AE8H           ;Renew all pointers in program
                                              ;text, active command mode


; Error at VERIFY

  02C8A 21A52C         LD     HL,2CA5H        ;HL -> 'BAD'
  02C8D CD7935         CALL   3579H           ;Print text and give tone
  02C90 C3181A         JP     1A18H           ;Back to active command mode


; Filename not found

  02C93 322644         LD     (4426H),A       ;Print actual filename found
  02C96 0603           LD     B,03H           ;Seach for end of program
                                              ;(3 times 00H)
  02C98 CDED01         CALL   01EDH           ;Read byte
  02C9B B7             OR     A               ;00H found ?
  02C9C 20F8           JR     NZ,2C96H        ;No: next byte

  02C9E 10F8           DJNZ   2C98H           ;Counter - 1: read next byte

  02CA0 00             NOP                    ;--
  02CA1 00             NOP
  02CA2 00             NOP
  02CA3 18A2           JR     2C47H           ;Retry CLOAD


; Text 'BAD'

  02CA5 424144         DEFB   'BAD'
  02CA8 0D             DEFB   0DH
  02CA9 00             DEFB   00H             ;End of text


; X = PEEK ( X )
; --------------

  02CAA CD7F0A         CALL   0A7FH           ;HL = X = CINT (X) = address
  02CAD 7E             LD     A,(HL)          ;A = memory value
  02CAE C3F827         JP     27F8H           ;Save A to X as INT
```

```
; POKE statement
; --------------

  02CB1 CD022B          CALL    2B02H           ;DE = address
  02CB4 D5              PUSH    DE              ;Save address
  02CB5 CF              RST     08H             ;Next byte must be
  02CB6 2C              DEFB    ','             ;a comma
  02CB7 CD1C2B          CALL    2B1CH           ;Get poke value
  02CBA D1              POP     DE              ;Get address
  02CBB 12              LD      (DE),A          ;Store value in memory
  02CBC C9              RET


; PRINT USING
; -----------

  02CBD CD3823          CALL    2338H           ;Get format string
  02CC0 CDF40A          CALL    0AF4H           ;?TM Error if no string found
  02CC3 CF              RST     08H             ;Next character must
  02CC4 3B              DEFB    ';'             ;be ';'
  02CC5 EB              EX      DE,HL           ;DE = PTP
  02CC6 2A2141          LD      HL,(4121H)      ;HL -> string vector
  02CC9 1808            JR      2CD3H           ;Continue at 2CD3H

; Re-entry in case multiple nuerical values have to be printed using the same
; format string

  02CCB 3ADE40          LD      A,(40DEH)       ;A = character following
                                                ;separator
  02CCE B7              OR      A               ;Separator followed by a
                                                ;variable ?
  02CCF 280C            JR      Z,2CDDH         ;No: ?FC Error
                                                ;Yes:
  02CD1 D1              POP     DE              ;Restore vector address of
                                                ;format string
  02CD2 EB              EX      DE,HL           ;HL -> vector, DE = PTP

; Execute PRINT USING
;
; I: DE =  PTP on variable to be printed
;    HL -> format string vector

  02CD3 E5              PUSH    HL              ;Save vector address
  02CD4 AF              XOR     A               ;A = 0
  02CD5 32DE40          LD      (40DEH),A       ;Clear character
  02CD8 BA              CP      D               ;C-flag = 1, Z-flag = 0
  02CD9 F5              PUSH    AF              ;Save AF
  02CDA D5              PUSH    DE              ;Save PTP
  02CDB 46              LD      B,(HL)          ;B = format string length
  02CDC B0              OR      B               ;Null string ?
  02CDD CA4A1E          JP      Z,1E4AH         ;Yes: ?FC Error
                                                ;No:
  02CE0 23              INC     HL              ;Vector address + 1
  02CE1 4E              LD      C,(HL)          ;HL = string pointer
  02CE2 23              INC     HL
  02CE3 66              LD      H,(HL)
  02CE4 69              LD      L,C
  02CE5 181C            JR      2D03H           ;Continue at 2D03H

; '%' found
;
; I: B  = remaining string length
;    HL = pointer on string (on next character following '%')
; O: C  = number of spaces + 2 found between both '%' characters

  02CE7 58              LD      E,B             ;Save counter
  02CE8 E5              PUSH    HL              ;Save pointer
  02CE9 0E02            LD      C,02H           ;C = counter for spaces
                                                ;(2 for the 2 '%' delimiters)
  02CEB 7E              LD      A,(HL)          ;Get next character
  02CEC 23              INC     HL              ;Pointer + 1
```

```
  02CED FE25            CP      '%'             ;2nd '%' found ?
  02CEF CA172E          JP      Z,2E17H         ;Yes: done, continue at 2E17H
                                                ;No:
  02CF2 FE20            CP      20H             ;Space?
  02CF4 2003            JR      NZ,2CF9H        ;No: use '%' as text character
                                                ;(only spaces are allowed
                                                ;between '%' delimiters)
  02CF6 0C              INC     C               ;Counter + 1
  02CF7 10F2            DJNZ    2CEBH           ;Get next character

  02CF9 E1              POP     HL              ;Restore string pointer in case
                                                ;no 2nd '%' was found
  02CFA 43              LD      B,E             ;Restore previous counter value
  02CFB 3E25            LD      A,25H           ;and use '%' as text character

; The character found is not a formatting character but a part of a text to be
; printed

  02CFD CD492E          CALL    2E49H           ;If D <> 0 then print '+'
  02D00 CD2A03          CALL    032AH           ;Print character

; Process format string
;
; I: (SP - 4) -> format string vector
;    (SP - 2) =  AF (A = 00H, C-flag = 1, Z-flag = 0. see 2CD9H)
;    (SP)     =  PTP
;     B       =  string length (remaining length)
;     HL      =  string pointer

  02D03 AF              XOR     A               ;A = 00H
  02D04 5F              LD      E,A             ;E = number of positions before
                                                ;the decimal point = 0
  02D05 57              LD      D,A             ;Format byte = 0
                                                ;(For composition of format
                                                ;byte see 0FBEH)
  02D06 CD492E          CALL    2E49H           ;If D <> 0 then print '+'
  02D09 57              LD      D,A             ;D = format byte
  02D0A 7E              LD      A,(HL)          ;Get next character
  02D0B 23              INC     HL              ;Pointer + 1
  02D0C FE21            CP      '!'             ;'!' found ?
  02D0E CA142E          JP      Z,2E14H         ;Yes: continue at 2E14H

  02D11 FE23            CP      '#'             ;'#' found ?
  02D13 2837            JR      Z,2D4CH         ;Yes: continue at 2D4CH

  02D15 05              DEC     B               ;Counter - 1
                                                ;String end reached ?
  02D16 CAFE2D          JP      Z,2DFEH         ;Yes: continue at 2DFEH

  02D19 FE2B            CP      '+'             ;'+' found ?
  02D1B 3E08            LD      A,08H           ;Set bit 3 of format byte
  02D1D 28E7            JR      Z,2D06H         ;Yes: continue at 2D06H

  02D1F 2B              DEC     HL              ;Pointer - 1
  02D20 7E              LD      A,(HL)          ;Get character again
  02D21 23              INC     HL              ;Pointer + 1
  02D22 FE2E            CP      '.'             ;'.' found ?
  02D24 2840            JR      Z,2D66H         ;Yes: continue at 2D66H

  02D26 FE25            CP      '%'             ;'%' found ?
  02D28 28BD            JR      Z,2CE7H         ;Yes: continue at 2CE7H

  02D2A BE              CP      (HL)            ;Same character twice ?
  02D2B 20D0            JR      NZ,2CFDH        ;No: character not recognized
                                                ;Yes:
  02D2D FE24            CP      '$'             ;'$$' found ?
  02D2F 2814            JR      Z,2D45H         ;Yes: continue at 2D45H

  02D31 FE2A            CP      '*'             ;'**' found ?
  02D33 20C8            JR      NZ,2CFDH        ;No: character not recognized
```

```
; '**' found

  02D35 78                LD      A,B             ;A = remaining length
  02D36 FE02              CP      02H             ;Less than 2 bytes remaining?
  02D38 23                INC     HL              ;Pointer + 1
  02D39 3803              JR      C,2D3EH         ;Yes: continue at 2D3EH

  02D3B 7E                LD      A,(HL)          ;Get next character
  02D3C FE24              CP      '$'             ;Is it a '$' ?
  02D3E 3E20              LD      A,20H           ;Set bit 5 (for '*' output)
  02D40 2007              JR      NZ,2D49H        ;No: only '**' found
                                                  ;Yes:
  02D42 05                DEC     B               ;'**$' found, counter - 1
  02D43 1C                INC     E               ;Positions before decimal
                                                  ;point + 1
  02D44 FEAF              CP      AFH             ;--

; '$$' found

* 02D45   AF              XOR     A               ;A = 0
  02D46 C610              ADD     A,10H           ;Set bit 4 ('$' in front)
  02D48 23                INC     HL              ;Pointer + 1
  02D49 1C                INC     E               ;Positions before decimal
                                                  ;point + 1
  02D4A 82                ADD     A,D             ;update A with format byte
  02D4B 57                LD      D,A             ;D = format byte

; '#' found

  02D4C 1C                INC     E               ;Positions before decimal
                                                  ;point + 1
  02D4D 0E00              LD      C,00H           ;Clear counter for text output
  02D4F 05                DEC     B               ;String counter - 1
                                                  ;Last character ?
  02D50 2847              JR      Z,2D99H         ;Yes: continue at 2D99H

  02D52 7E                LD      A,(HL)          ;Get next character
  02D53 23                INC     HL              ;Pointer + 1
  02D54 FE2E              CP      '.'             ;Decimal point found ?
  02D56 2818              JR      Z,2D70H         ;Yes: continue at 2D70H

  02D58 FE23              CP      '#'             ;'#' found ?
  02D5A 28F0              JR      Z,2D4CH         ;Yes: back to 2D4CH

  02D5C FE2C              CP      ','             ;',' found ?
  02D5E 201A              JR      NZ,2D7AH        ;No: continue at 2D7AH

; '.' found

  02D60 7A                LD      A,D             ;Set bit 6 of format byte
  02D61 F640              OR      40H
  02D63 57                LD      D,A             ;D = format byte
  02D64 18E6              JR      2D4CH           ;Get next character

; '.' in front of '#' found

  02D66 7E                LD      A,(HL)          ;A = next character
  02D67 FE23              CP      '#'             ;Next character a '#' ?
  02D69 3E2E              LD      A,2EH           ;A = '.'
  02D6B 2090              JR      NZ,2CFDH        ;No: use '.' as text character
                                                  ;Yes:
  02D6D 0E01              LD      C,01H           ;Set counter for positions
                                                  ;behind the decimal point to 1
  02D6F 23                INC     HL              ;Pointer + 1

; '.' behind '#' found

  02D70 0C                INC     C               ;Positions behind decimal
                                                  ;point + 1
  02D71 05                DEC     B               ;Counter - 1
                                                  ;String end ?
```

```
  02D72 2825            JR      Z,2D99H         ;Yes: continue at 2D99H

  02D74 7E              LD      A,(HL)          ;Get next character
  02D75 23              INC     HL              ;Pointer + 1
  02D76 FE23            CP      '#'             ;'#' ?
  02D78 28F6            JR      Z,2D70H         ;Yes: Positions behind decimal
                                                ;point + 1
; Exponential format specified ?
; (4 consecutive times arrow up)

  02D7A D5              PUSH    DE              ;Save format byte and counter
  02D7B 11972D          LD      DE,2D97H        ;Set new RET address
  02D7E D5              PUSH    DE              ;to 2D97H
  02D7F 54              LD      D,H             ;DE = string pointer
  02D80 5D              LD      E,L
  02D81 FE5B            CP      '['             ;Arrow up found ?
  02D83 C0              RET     NZ              ;No: RET to 2D79H

  02D84 BE              CP      (HL)            ;2 times arrow up ?
  02D85 C0              RET     NZ              ;No: RET to 2D79H

  02D86 23              INC     HL              ;Pointer + 1
  02D87 BE              CP      (HL)            ;3 times arrow up ?
  02D88 C0              RET     NZ              ;No: RET to 2D79H

  02D89 23              INC     HL              ;Pointer + 1
  02D8A BE              CP      (HL)            ;4 times arrow up ?
  02D8B C0              RET     NZ              ;No: RET to 2D79H

  02D8C 23              INC     HL              ;Pointer + 1
  02D8D 78              LD      A,B             ;A = string counter
  02D8E D604            SUB     04H             ;Subtract 4, overflow ?
  02D90 D8              RET     C               ;Yes: the 4 arrow ups are not
                                                ;all part of the format string

; 4 consecutive arrow ups found: print number in exponential format

  02D91 D1              POP     DE              ;Remove RET address (2D97H)
  02D92 D1              POP     DE              ;Restore format byte and
                                                ;number of positions before
                                                ;the decimal point
  02D93 47              LD      B,A             ;B = string counter
  02D94 14              INC     D               ;Set bit 0 (exponential format)
  02D95 23              INC     HL              ;Pointer + 1
  02D96 CAEBD1          JP      Z,0D1EBH        ;Jump is never executed because
                                                ;Z-flag <> 0 (because of 2D94H)
; No exponential format output

* 02D97   EB            EX      DE,HL           ;HL = old string pointer
* 02D98     D1          POP     DE              ;Restore format byte and number
                                                ;positions before decimal point
; Last string character was '.' or '#'

  02D99 7A              LD      A,D             ;A = format byte
  02D9A 2B              DEC     HL              ;Pointer - 1
  02D9B 1C              INC     E               ;Positions before decimal
                                                ;point + 1
  02D9C E608            AND     08H             ;Print positive sign ?
  02D9E 2015            JR      NZ,2DB5H        ;Yes: continue at 2DB5H
                                                ;No:
  02DA0 1D              DEC     E               ;Positions before decimal
                                                ;point - 1
  02DA1 78              LD      A,B             ;Any string characters left ?
  02DA2 B7              OR      A
  02DA3 2810            JR      Z,2DB5H         ;No: continue at 2DB5H

  02DA5 7E              LD      A,(HL)          ;A = next character
  02DA6 D62D            SUB     2DH             ;'-' found ?
  02DA8 2806            JR      Z,2DB0H         ;Yes: continue at 2DB0H

  02DAA FEFE            CP      0FEH            ;'+' found ?
```

```
  02DAC 2007            JR      NZ,2DB5H        ;No: continue at 2DB5H

  02DAE 3E08            LD      A,08H           ;Set bit 3 (print sign)
  02DB0 C604            ADD     A,04H           ;Set bit 2 (behind number)
  02DB2 82              ADD     A,D             ;Update with format byte
  02DB3 57              LD      D,A             ;D = format byte
  02DB4 05              DEC     B               ;String counter - 1

; Processing of format string ready
;
; I: B = string counter
;    C = number of positions behind decimal point + 1 (for decimal point)
;    D = format byte
;    E = number of positions in front of decimal point

  02DB5 E1              POP     HL              ;Restore PTP
  02DB6 F1              POP     AF              ;Restore flags (see 2CD9H)
  02DB7 2850            JR      Z,2E09H         ;Done when Z-flag = 0

  02DB9 C5              PUSH    BC              ;Save registers
  02DBA D5              PUSH    DE
  02DBB CD3723          CALL    2337H           ;Get value to be printed
  02DBE D1              POP     DE              ;Restore registers
  02DBF C1              POP     BC
  02DC0 C5              PUSH    BC              ;Save string counter
  02DC1 E5              PUSH    HL              ;Save PTP
  02DC2 43              LD      B,E             ;B = number of positions in
                                                ;front of decimal point
  02DC3 78              LD      A,B             ;A = B
  02DC4 81              ADD     A,C             ;Add number of positions behind
                                                ;the decimal point
  02DC5 FE19            CP      19H             ;More than 24 positions ?
  02DC7 D24A1E          JP      NC,1E4AH        ;Yes: ?FC Error, the maximum
                                                ;number of positions is 24
                                                ;( 1 for sign,
                                                ; 17 for DBL format
                                                ;  1 for decimal point
                                                ;  4 for exponent
                                                ;  1 for sign following number)
  02DCA 7A              LD      A,D             ;A = format byte
  02DCB F680            OR      80H             ;Execute formatting
                                                ;(bit 7 set ?)
  02DCD CDBE0F          CALL    0FBEH           ;Store number in X as formatted
                                                ;string at 4130H. HL -> string
  02DD0 CDA728          CALL    28A7H           ;Print string
  02DD3 E1              POP     HL              ;Restore PTP
  02DD4 2B              DEC     HL              ;Adjust PTP
  02DD5 D7              RST     10H             ;Get next character
  02DD6 37              SCF                     ;C-flag = 1
                                                ;End of assignment reached ?
  02DD7 280D            JR      Z,2DE6H         ;Yes: continue at 2DE6H

  02DD9 32DE40          LD      (40DEH),A       ;Store next character
                                                ;(see 2CCBH)
  02DDC FE3B            CP      3BH             ;';' found ?
  02DDE 2805            JR      Z,2DE5H         ;Yes: character ok

  02DE0 FE2C            CP      2CH             ;',' found ?
  02DE2 C29719          JP      NZ,1997H        ;No: ?SN Error

  02DE5 D7              RST     10H             ;Adjust PTP
  02DE6 C1              POP     BC              ;Restore string counter
  02DE7 EB              EX      DE,HL           ;DE = PTP
  02DE8 E1              POP     HL              ;HL -> format string vector
  02DE9 E5              PUSH    HL              ;Save vector address
  02DEA F5              PUSH    AF              ;Save next character
  02DEB D5              PUSH    DE              ;Save PTP
  02DEC 7E              LD      A,(HL)          ;A = format string length
  02DED 90              SUB     B               ;Subtract remaining length
  02DEE 23              INC     HL              ;Vector address + 1
  02DEF 4E              LD      C,(HL)          ;HL -> format string
```

```
  02DF0 23              INC     HL
  02DF1 66              LD      H,(HL)
  02DF2 69              LD      L,C
  02DF3 1600            LD      D,00H           ;DE = offset for next format
  02DF5 5F              LD      E,A             ;string (format character
                                                ;for next value to print)
  02DF6 19              ADD     HL,DE           ;Add offset. HL now points
                                                ;to next format indication
  02DF7 78              LD      A,B             ;A = remaining length
  02DF8 B7              OR      A               ;Length = 0 ?
  02DF9 C2032D          JP      NZ,2D03H        ;No: process format string
                                                ;from (HL) onwards
  02DFC 1806            JR      2E04H           ;Yes: use old format string
                                                ;also for the next value
; End of string found

  02DFE CD492E          CALL    2E49H           ;If D <> 0 then print '+'
  02E01 CD2A03          CALL    032AH           ;Print last character
  02E04 E1              POP     HL              ;Restore PTP
  02E05 F1              POP     AF              ;Restore flags/next character
                                                ;(see 2DEAH)
  02E06 C2CB2C          JP      NZ,2CCBH        ;Process next number with same
                                                ;format string
  02E09 DCFE20          CALL    C,20FEH         ;End PRINT if end of assigment
                                                ;is reached (see 2DD6H)
  02E0C E3              EX      (SP),HL         ;Save PTP, restore vector
                                                ;address
  02E0D CDDD29          CALL    29DDH           ;Remove format string from
                                                ;string table and string space
  02E10 E1              POP     HL              ;Restore PTP
  02E11 C36921          JP      2169H           ;Output back to screen

; '!' found

  02E14 0E01            LD      C,01H           ;Print 1 text character
  02E16 3EF1            LD      A,0F1H          ;--

; 2nd '%' character found
; C = number of characters to be printer

* 02E17   F1            POP     AF              ;Remove string pointer from
                                                ;stack
  02E18 05              DEC     B               ;Counter - 1
  02E19 CD492E          CALL    2E49H           ;If D <> 0 then print '+'
  02E1C E1              POP     HL              ;Restore PTP
  02E1D F1              POP     AF              ;Restore flags: done ?
  02E1E 28E9            JR      Z,2E09H         ;Yes: end PRINT operation
                                                ;No:
  02E20 C5              PUSH    BC              ;Save string counter
  02E21 CD3723          CALL    2337H           ;Get string to be printed
  02E24 CDF40A          CALL    0AF4H           ;?TM Error if not a string
  02E27 C1              POP     BC              ;Restore string counter
  02E28 C5              PUSH    BC              ;Save string counter
  02E29 E5              PUSH    HL              ;Save PTP
  02E2A 2A2141          LD      HL,(4121H)      ;HL -> vector of string to be
                                                ;printed
  02E2D 41              LD      B,C             ;B = number of characters to
                                                ;be printed
  02E2E 0E00            LD      C,00H           ;C = 0: ouput is started at
                                                ;start of string
  02E30 C5              PUSH    BC              ;Save counters
  02E31 CD682A          CALL    2A68H           ;Get characters to be printed
                                                ;via LEFT$
  02E34 CDAA28          CALL    28AAH           ;Print characters
  02E37 2A2141          LD      HL,(4121H)      ;HL -> vector of printed string
  02E3A F1              POP     AF              ;A = number of printed chars
  02E3B 96              SUB     (HL)            ;- total string length
  02E3C 47              LD      B,A             ;B = remaining string length
  02E3D 3E20            LD      A,20H           ;A = space
  02E3F 04              INC     B               ;Counter + 1
  02E40 05              DEC     B               ;Counter reached 0 ?
```

```
  02E41 CAD32D           JP      Z,2DD3H          ;Yes: process next argument
                                                  ;No:
  02E44 CD2A03           CALL    032AH            ;Replace missing characters
                                                  ;by spaces
  02E47 18F7             JR      2E40H            ;next character


; If D <> 0 then print '+'

  02E49 F5               PUSH    AF               ;Save AF
  02E4A 7A               LD      A,D              ;D <> 0 ?
  02E4B B7               OR      A
  02E4C 3E2B             LD      A,2BH            ;A = '+'
  02E4E C42A03           CALL    NZ,032AH         ;Yes: print '+'
  02E51 F1               POP     AF               ;Restore AF
  02E52 C9               RET


; Entry for EDIT after ?SN Error

  02E53 329A40           LD      (409AH),A        ;Clear last error code
  02E56 2AEA40           LD      HL,(40EAH)       ;HL = ERL
  02E59 B4               OR      H                ;ERL = 65535 ?
  02E5A A5               AND     L                ;(Syntax error in active
  02E5B 3C               INC     A                ;command mmode)
  02E5C EB               EX      DE,HL            ;DE = ERL
  02E5D C8               RET     Z                ;Yes: no EDIT has to be done
                                                  ;No:
  02E5E 1804             JR      2E64H            ;Call EDIT with LN = DE


; EDIT statement
; --------------

  02E60 CD4F1E           CALL    1E4FH            ;Load line number into DE
  02E63 C0               RET     NZ               ;?SN Error ?

  02E64 E1               POP     HL               ;Remove RET address from stack
  02E65 EB               EX      DE,HL            ;HL = line number
  02E66 22EC40           LD      (40ECH),HL       ;Save as '.'
  02E69 EB               EX      DE,HL            ;DE = line number
  02E6A CD2C1B           CALL    1B2CH            ;Search for line DE
  02E6D D2D91E           JP      NC,1ED9H         ;?UL Error in case the line
                                                  ;does not exist
  02E70 60               LD      H,B              ;HL = line pointer
  02E71 69               LD      L,C
  02E72 23               INC     HL               ;HL + 2
  02E73 23               INC     HL               ;BC = line number
  02E74 4E               LD      C,(HL)
  02E75 23               INC     HL
  02E76 46               LD      B,(HL)
  02E77 23               INC     HL
  02E78 C5               PUSH    BC               ;Save LN
  02E79 CD7E2B           CALL    2B7EH            ;Decode line from (HL) onwards
                                                  ;and store in line buffer
  02E7C E1               POP     HL               ;Restore LN
  02E7D E5               PUSH    HL               ;And save it again
  02E7E CDAF0F           CALL    0FAFH            ;Print HL as decimal number
  02E81 3E20             LD      A,20H            ;A = ' '
  02E83 CD2A03           CALL    032AH            ;Print it
  02E86 2AA740           LD      HL,(40A7H)       ;HL -> line buffer
  02E89 3E0E             LD      A,0EH            ;Cursor on
  02E8B CD2A03           CALL    032AH
  02E8E E5               PUSH    HL               ;Save pointer
  02E8F 0EFF             LD      C,0FFH           ;C = counter (set to -1 because
                                                  ;of following INC C)
  02E91 0C               INC     C                ;Counter + 1
  02E92 7E               LD      A,(HL)           ;A = next characater
  02E93 B7               OR      A                ;End of line reached
  02E94 23               INC     HL               ;Pointer + 1
  02E95 20FA             JR      NZ,2E91H         ;No: continue counting
```

```
                                                    ;Yes:
  02E97 E1              POP     HL                  ;Restore pointer
  02E98 47              LD      B,A                 ;B = 00H (number of already
                                                    ;printed characaters)
                                                    ;C = length of line
  02E99 1600            LD      D,00H               ;D = 0 (repetition counter)
  02E9B CD8403          CALL    0384H               ;Get character from keyboard
  02E9E D630            SUB     30H                 ;Can is be a digit ?
  02EA0 380E            JR      C,2EB0H             ;No: continue at 2EB0H
                                                    ;Yes:
  02EA2 FE0A            CP      0AH                 ;Digit ?
  02EA4 300A            JR      NC,2EB0H            ;No: continue at 2EB0H
                                                    ;Yes:
  02EA6 5F              LD      E,A                 ;E = number value (00H-09H)
  02EA7 7A              LD      A,D                 ;A = last value
  02EA8 07              RLCA                        ;*2
  02EA9 07              RLCA                        ;*2 (*4 in total)
  02EAA 82              ADD     A,D                 ;+value (*5 in total)
  02EAB 07              RLCA                        ;*2 (*10 in total)
  02EAC 83              ADD     A,E                 ;Add next decimal position
  02EAD 57              LD      D,A                 ;D = counter
  02EAE 18EB            JR      2E9BH               ;Get next character

; No digit entered
; D = repetition counter
; A = ASCII code of entered character - 30H (!)

  02EB0 E5              PUSH    HL                  ;Save pointer
  02EB1 21992E          LD      HL,2E99H            ;Set new RET address
  02EB4 E3              EX      (SP),HL             ;to 2E99H and restore pointer
  02EB5 15              DEC     D                   ;Repetition required ?
  02EB6 14              INC     D                   ;Is D <> 0 ?
  02EB7 C2BB2E          JP      NZ,2EBBH            ;Yes: leave D as it is
                                                    ;No:
  02EBA 14              INC     D                   ;Set D to 1 (execute required
                                                    ;function at least once)
  02EBB FED8            CP      0D8H                ;08H = Backspace ?
  02EBD CAD22F          JP      Z,2FD2H             ;Yes: continue at 2FD2H

  02EC0 FEDD            CP      0DDH                ;0DH = RETURN ?
  02EC2 CAE02F          JP      Z,2FE0H             ;Yes: continue at 2FE0H

  02EC5 FEF0            CP      0F0H                ;20H = space ?
  02EC7 2841            JR      Z,2F0AH             ;Yes: continue at 2F0AH

  02EC9 FE31            CP      31H                 ;Lower case (char > 60H) ?
  02ECB 3802            JR      C,2ECFH             ;No: character ok.
                                                    ;Yes:
  02ECD D620            SUB     20H                 ;Convert to upper case
  02ECF FE21            CP      21H                 ;51H = 'Q' ?
  02ED1 CAF62F          JP      Z,2FF6H             ;Yes: continue at 2FF6H

  02ED4 FE1C            CP      1CH                 ;4CH = 'L' ?
  02ED6 CA402F          JP      Z,2F40H             ;Yes: continue at 2F40H

  02ED9 FE23            CP      23H                 ;53H = 'S' ?
  02EDB 283F            JR      Z,2F1CH             ;Yes: continue at 2F1CH

  02EDD FE19            CP      19H                 ;49H = 'I' ?
  02EDF CA7D2F          JP      Z,2F7DH             ;Yes: continue at 2F7DH

  02EE2 FE14            CP      14H                 ;44H = 'D' ?
  02EE4 CA4A2F          JP      Z,2F4AH             ;Yes: continue at 2F4AH

  02EE7 FE13            CP      13H                 ;43H = 'C' ?
  02EE9 CA652F          JP      Z,2F65H             ;Yes: continue at 2F65H

  02EEC FE15            CP      15H                 ;45H = 'E' ?
  02EEE CAE32F          JP      Z,2FE3H             ;Yes: continue at 2FE3H

  02EF1 FE28            CP      28H                 ;58H = 'X' ?
```

```
  02EF3 CA782F          JP      Z,2F78H         ;Yes: continue at 2F78H

  02EF6 FE1B            CP      1BH             ;4BH = 'K' ?
  02EF8 281C            JR      Z,2F16H         ;Yes: continue at 2F16H

  02EFA FE18            CP      18H             ;48H = 'H' ?
  02EFC CA752F          JP      Z,2F75H         ;Yes: continue at 2F75H

  02EFF FE11            CP      11H             ;41H = 'A' ?
  02F01 C0              RET     NZ              ;No: back to 2E99H

; 'A': start again

  02F02 C1              POP     BC              ;Remove RET address
  02F03 D1              POP     DE              ;Restore LN
  02F04 CDFE20          CALL    20FEH           ;Start new line on screen
  02F07 C3652E          JP      2E65H           ;Restart EDIT

; Space bar: print next character

  02F0A 7E              LD      A,(HL)          ;Get next character
  02F0B B7              OR      A               ;End of line reached ?
  02F0C C8              RET     Z               ;Yes: done, return

  02F0D 04              INC     B               ;Counter + 1
  02F0E CD2A03          CALL    032AH           ;Print character
  02F11 23              INC     HL              ;Pointer + 1
  02F12 15              DEC     D               ;Repeat ?
  02F13 20F5            JR      NZ,2F0AH        ;Yes: next character

  02F15 C9              RET

; 'K': delete line up to character entered

  02F16 E5              PUSH    HL              ;Save pointer
  02F17 215F2F          LD      HL,2F5FH        ;Set new RET address to 2F5FH
                                                ;(to print '!' at the end)
  02F1A E3              EX      (SP),HL         ;Restore pointer
  02F1B 37              SCF                     ;C-flag = 0 (indicator for
                                                ;delete character)
; 'S': search for character entered

  02F1C F5              PUSH    AF              ;Save flags
  02F1D CD8403          CALL    0384H           ;Get character
  02F20 5F              LD      E,A             ;E = character
  02F21 F1              POP     AF              ;Restore flags
  02F22 F5              PUSH    AF              ;And save them again
  02F23 DC5F2F          CALL    C,2F5FH         ;Print '!' for 'K' operation
  02F26 7E              LD      A,(HL)          ;A = next character
  02F27 B7              OR      A               ;End of line reached ?
  02F28 CA3E2F          JP      Z,2F3EH         ;Yes: done.
                                                ;No:
  02F2B CD2A03          CALL    032AH           ;Print character
  02F2E F1              POP     AF              ;Restore flags
  02F2F F5              PUSH    AF              ;And save them again
                                                ;'K' operation ?
  02F30 DCA12F          CALL    C,2FA1H         ;Yes: Delete character
  02F33 3802            JR      C,2F37H         ;Yes: Skip next operation
                                                ;(counter and pointer are
                                                ;adjusted at 2FA1H)
  02F35 23              INC     HL              ;Pointer + 1
  02F36 04              INC     B               ;Counter + 1
  02F37 7E              LD      A,(HL)          ;A = next character
  02F38 BB              CP      E               ;= searched character ?
  02F39 20EB            JR      NZ,2F26H        ;No: next character
                                                ;Yes:
  02F3B 15              DEC     D               ;Repeat done ?
  02F3C 20E8            JR      NZ,2F26H        ;No: next character
                                                ;Yes:
  02F3E F1              POP     AF              ;Restore flags
  02F3F C9              RET
```

```
; 'L': list line and start again

  02F40 CD752B          CALL    2B75H           ;Print line
  02F43 CDFE20          CALL    20FEH           ;Start new line on screen
  02F46 C1              POP     BC              ;Remove RET address
  02F47 C37C2E          JP      2E7CH           ;Restart EDIT

; 'D': Delete character

  02F4A 7E              LD      A,(HL)          ;A = character
  02F4B B7              OR      A               ;End of line reached?
  02F4C C8              RET     Z               ;Yes: done, return.
                                                ;No:
  02F4D 3E21            LD      A,21H           ;A = '!'
  02F4F CD2A03          CALL    032AH           ;Print it
  02F52 7E              LD      A,(HL)          ;A = character
  02F53 B7              OR      A               ;End of line ?
  02F54 2809            JR      Z,2F5FH         ;Yes: done.
                                                ;No:
  02F56 CD2A03          CALL    032AH           ;Print character
  02F59 CDA12F          CALL    2FA1H           ;and delete
  02F5C 15              DEC     D               ;Repeat ?
  02F5D 20F3            JR      NZ,2F52H        ;Yes: next character

; Print '!'

  02F5F 3E21            LD      A,21H           ;A = '!'
  02F61 CD2A03          CALL    032AH           ;Print it
  02F64 C9              RET

; 'C': change character

  02F65 7E              LD      A,(HL)          ;A = character
  02F66 B7              OR      A               ;End of line reached ?
  02F67 C8              RET     Z               ;Yes : done, return.

  02F68 CD8403          CALL    0384H           ;Get new character
  02F6B 77              LD      (HL),A          ;Use it
  02F6C CD2A03          CALL    032AH           ;And print it
  02F6F 23              INC     HL              ;Pointer + 1
  02F70 04              INC     B               ;Counter + 1
  02F71 15              DEC     D               ;Repeat ?
  02F72 20F1            JR      NZ,2F65H        ;Yes: next character
                                                ;No:
  02F74 C9              RET                     ;Done.

; 'H': Hack off rest of the line and jump to 'I'

  02F75 3600            LD      (HL),00H        ;Set end of line at current
                                                ;position
  02F77 48              LD      C,B             ;C = number of already printed
                                                ;characters = new line length
; 'X': print rest of the line and jump to 'I'

  02F78 16FF            LD      D,0FFH          ;Execute 255 times the space
  02F7A CD0A2F          CALL    2F0AH           ;bar function to print the
                                                ;remaining part of the line
; 'I': insert new characters

  02F7D CD8403          CALL    0384H           ;Get new character
  02F80 B7              OR      A               ;(why ?)
  02F81 CA7D2F          JP      Z,2F7DH         ;(see 0384H)

  02F84 FE08            CP      08H             ;Arrow left ?
  02F86 280A            JR      Z,2F92H         ;Yes: delete character

  02F88 FE0D            CP      0DH             ;RETURN ?
  02F8A CAE02F          JP      Z,2FE0H         ;Yes: execute RETURN function

  02F8D FE1B            CP      1BH             ;SHIFT+arrow up ? (End I-func)
```

```
  02F8F C8               RET     Z                ;Yes: done, return
                                                  ;No:
  02F90 201E             JR      NZ,2FB0H         ;Insert character

; Arrow left: Delete character in I-function

  02F92 3E08             LD      A,08H            ;A = ASCII code of delete
                                                  ;character = backspace
  02F94 05               DEC     B                ;Any character already printed?
  02F95 04               INC     B                ;(is there anything to delete)
  02F96 281F             JR      Z,2FB7H          ;No: done
                                                  ;Yes:
  02F98 CD2A03           CALL    032AH            ;Delete character from screen
  02F9B 2B               DEC     HL               ;Pointer - 1
  02F9C 05               DEC     B                ;Counter - 1
  02F9D 117D2F           LD      DE,2F7DH         ;Set new RET address
  02FA0 D5               PUSH    DE               ;to 2F7DH

; Delete character at (HL)

  02FA1 E5               PUSH    HL               ;Save pointer
  02FA2 0D               DEC     C                ;Length of line - 1
  02FA3 7E               LD      A,(HL)           ;Get character
  02FA4 B7               OR      A                ;End of line reached ?
  02FA5 37               SCF                      ;C-flag = 1
  02FA6 CA9008           JP      Z,0890H          ;Yes: restore pointer and done
                                                  ;No:
  02FA9 23               INC     HL               ;Pointer + 1
  02FAA 7E               LD      A,(HL)           ;Get next character
  02FAB 2B               DEC     HL               ;pointer - 1
  02FAC 77               LD      (HL),A           ;And store at current position
  02FAD 23               INC     HL               ;Pointer + 1
  02FAE 18F3             JR      2FA3H            ;Shift line until end of line
                                                  ;is reached
; Insert character  at (HL)

  02FB0 F5               PUSH    AF               ;Save character
  02FB1 79               LD      A,C              ;A = length of line
  02FB2 FEFF             CP      0FFH             ;Maximum length reached ?
  02FB4 3803             JR      C,2FB9H          ;No: insert character
                                                  ;Yes:
  02FB6 F1               POP     AF               ;restore character
  02FB7 18C4             JR      2F7DH            ;Back to the 'I'-function

; Insert character

  02FB9 90               SUB     B                ;A = length of line - number
                                                  ;of characters already printed
                                                  ;= remaining length
  02FBA 0C               INC     C                ;Length of line + 1
  02FBB 04               INC     B                ;Printed characters + 1
  02FBC C5               PUSH    BC               ;Save counter
  02FBD EB               EX      DE,HL            ;DE = pointer
  02FBE 6F               LD      L,A              ;HL = remaining length
  02FBF 2600             LD      H,00H
  02FC1 19               ADD     HL,DE            ;HL = current length +
                                                  ;remaining length = pointer to
                                                  ;the end of line
  02FC2 44               LD      B,H              ;BC = end pointer
  02FC3 4D               LD      C,L
  02FC4 23               INC     HL               ;HL = end pointer + 1
  02FC5 CD5819           CALL    1958H            ;Shift line
  02FC8 C1               POP     BC               ;Restore counter
  02FC9 F1               POP     AF               ;Restore character
  02FCA 77               LD      (HL),A           ;Insert character
  02FCB CD2A03           CALL    032AH            ;Print character
  02FCE 23               INC     HL               ;ointer + 1
  02FCF C37D2F           JP      2F7DH            ;Back to the 'I'-function

; Arrow left: delete character left from cursor
```

```
  02FD2 78              LD      A,B             ;A = number of printed
                                                ;characters (+ number of
                                                ;characters left from cursor)
  02FD3 B7              OR      A               ;Anything printed ?
  02FD4 C8              RET     Z               ;No: done, return
                                                ;Yes:
  02FD5 05              DEC     B               ;Counter - 1
  02FD6 2B              DEC     HL              ;Pointer - 1
  02FD7 3E08            LD      A,08H           ;Delete character
  02FD9 CD2A03          CALL    032AH           ;on screen
  02FDC 15              DEC     D               ;Repeat ?
  02FDD 20F3            JR      NZ,2FD2H        ;Yes: next character

  02FDF C9              RET

; RETURN: Print remaining part of line and end EDIT

  02FE0 CD752B          CALL    2B75H           ;Print remaining line

; 'E': end EDIT

  02FE3 CDFE20          CALL    20FEH           ;Start new line on screen
  02FE6 C1              POP     BC              ;Remove RET address
  02FE7 D1              POP     DE              ;Restore LN
  02FE8 7A              LD      A,D             ;LN = 65535 ?
  02FE9 A3              AND     E               ;(set Z-flag for 2FEFH,
  02FEA 3C              INC     A               ;see 1A71H)
  02FEB 2AA740          LD      HL,(40A7H)      ;HL = line buffer address
  02FEE 2B              DEC     HL              ;Pointer - 1
                                                ;LN = 65535 ?
  02FEF C8              RET     Z               ;Yes: done, return

  02FF0 37              SCF                     ;C-flag = 1
  02FF1 23              INC     HL              ;Pointer + 1
  02FF2 F5              PUSH    AF              ;Save flags
  02FF3 C3981A          JP      1A98H           ;Insert line at HL into
                                                ;program text
; 'Q': terminate edit, ignore changes

  02FF6 C1              POP     BC              ;Restore RET address
  02FF7 D1              POP     DE              ;Restore LN
  02FF8 C3191A          JP      1A19H           ;Back to active command mode

  02FFB DEC3            SBC     A,0C3H          ;--
  02FFD C344B2          JP      0B244H


; Keyboard input with FKEY evaluation
; (Is only used by line input (at 05D9H) and has
; therefor 05E3H as return address!!

  03000 C31534          JP      3415H           ;Continue at 3415H

; No Function Key pressed (start of 3458H)
; CTRL + number 1 to 8 pressed for colour change ?

  03003 E5              PUSH    HL              ;Save pointer
  03004 211840          LD      HL,4018H        ;HL = CTRL byte
  03007 CB7E            BIT     7,(HL)          ;CTRL pressed ?
  03009 2812            JR      Z,301DH         ;No: done
                                                ;Yes:
  0300B CBBE            RES     7,(HL)          ;Clear bit
  0300D FE31            CP      31H             ;A number pressed ?
  0300F 380C            JR      C,301DH         ;No: put character in A

  03011 FE39            CP      39H             ;Valid number ?
  03013 3008            JR      NC,301DH        ;No: done
                                                ;Yes:
  03015 D631            SUB     31H             ;A = Colour code 0 to 7
  03017 CD2136          CALL    3621H           ;Store new colour code and
                                                ;cursor on
```

```
  0301A E1               POP     HL               ;Restore pointer
  0301B 18E3             JR      3000H            ;Get new character

; Give character in A to input routine

  0301D E1               POP     HL               ;Restore pointer
  0301E C3E305           JP      05E3H            ;Return to line input routine

  03021 FF               RST     38H              ;--

; Graphic character (> 7FH) recognized

  03022 FEC0             CP      0C0H             ;This comparison was used in
                                                  ;a previous version to output
                                                  ;the characters 192-255
                                                  ;(C0H-FFH) as tab-function.
                                                  ;(output of 0 to 63 spaces)
                                                  ;In the current version all
                                                  ;values > 7FH are printed as
                                                  ;graphic characters.
  03024 C30531           JP      3105H            ;Print character

; Previously used Tab-function (not used)

  03027 D6C0             SUB     0C0H             ;A = value - 192 (= number of
                                                  ;spaces to print)
  03029 CA0831           JP      Z,3108H          ;A = 0: done

  0302C 47               LD      B,A              ;B = counter
  0302D 3E20             LD      A,' '            ;A = space
  0302F C5               PUSH    BC               ;Save counter
  03030 CD8431           CALL    3184H            ;Print character
  03033 C1               POP     BC               ;Restore counter
  03034 10F7             DJNZ    302DH            ;Next character
  03036 C30831           JP      3108H

; Move cursor one position to the right (= CHR$(25) )

  03039 23               INC     HL               ;Cursor address + 1
  0303A E5               PUSH    HL               ;Save cursor address
  0303B CD6531           CALL    3165H            ;HL -> actual line
  0303E EB               EX      DE,HL            ;DE = HL
  0303F E1               POP     HL               ;Restore cursor address
  03040 DF               RST     18H              ;Next line reached ?
  03041 C0               RET     NZ               ;No: done
                                                  ;Yes:
  03042 11D8FF           LD      DE,0FFD8H        ;DE = -40
  03045 19               ADD     HL,DE            ;Subtract 40 (length of 1
                                                  ;line) from new address.
                                                  ;The cursor is located at the
                                                  ;start of line again
  03046 C9               RET


; This routine loads the CRTC register pair A and A + 1 with HL:
; 16 bit load for CRTC register
;
; I: A = register number of lower register of register pair
;    HL = value to be written into CRTC

  03047 C5               PUSH    BC               ;Save BC
  03048 E5               PUSH    HL               ;Save HL
  03049 0602             LD      B,02H            ;Counter = 2

  0304B 0EFA             LD      C,0FAH           ;C = CRTC select register
  0304D ED79             OUT     (C),A            ;Select register A
  0304F 0C               INC     C                ;C = CRTC data register
  03050 ED61             OUT     (C),H            ;Write data to CRTC register
  03052 3C               INC     A                ;Next register
  03053 65               LD      H,L              ;H = L
  03054 10F5             DJNZ    304BH            ;Loop
```

```
  03056 E1                POP     HL                ;Restore HL
  03057 C1                POP     BC                ;Restore BC
  03058 C9                RET

; Cursor off (= CHR$(15) )

  03059 E5                PUSH    HL                ;Save address
  0305A 210720            LD      HL,2007H          ;Set H and L for CRTC
                                                    ;programming
  0305D 1804              JR      3063H             ;Continue at 3063H

; Cursor on (= CHR$(14) )

  0305F E5                PUSH    HL                ;Save address
  03060 2A1940            LD      HL,(4019H)        ;H and L = value for CRTC
                                                    ;programming

  03063 3E0A              LD      A,0AH             ;Select CRTC registers 10+11
  03065 CD4730            CALL    3047H             ;Write HL into CRTC
  03068 7C                LD      A,H               ;A = cursor start scan line
  03069 E1                POP     HL                ;Restore address
  0306A E5                PUSH    HL                ;Save registers
  0306B D5                PUSH    DE
  0306C FE20              CP      20H               ;Cursor off ?
  0306E C41436            CALL    NZ,3614H          ;No: set colour of new cursor

  03071 00                NOP                       ;--
  03072 3E0E              LD      A,0EH             ;Select CRTC registers 14+15
                                                    ;(Cursor position)
  03074 CD4730            CALL    3047H             ;Write address into CRTC
  03077 D1                POP     DE                ;Restore registers
  03078 E1                POP     HL
  03079 C9                RET


; Save registers and set colour
;
; I: A = character at (HL)
;    HL -> screen location in LGR screen memory

  0307A E5                PUSH    HL                ;Save registers
  0307B D5                PUSH    DE
  0307C F5                PUSH    AF
  0307D C3F835            JP      35F8H             ;Continue at 35F8H


; This routine updates the colour memory at position HL + DE with the current
; text colour set (see 35F8H)
;
; I: HL + DE -> Colour memory byte to be updated
; O: -

  03080 19                ADD     HL,DE             ;Set pointer into colour mem.
  03081 E5                PUSH    HL                ;Save colour memory pointer
  03082 219043            LD      HL,4390H          ;HL -> colour codes table
  03085 110000            LD      DE,0000H          ;DE = offset = 0;
  03088 3A2340            LD      A,(4023H)         ;Take current colour value
  0308B 5F                LD      E,A               ;Use it as offset in table
  0308C 19                ADD     HL,DE             ;Set pointer into table
  0308D 7E                LD      A,(HL)            ;A = colour code
  0308E E1                POP     HL                ;Restore colour memory pointer

  0308F 77                LD      (HL),A            ;Write value into colour mem.
  03090 F1                POP     AF                ;Restore registers
  03091 D1                POP     DE
  03092 E1                POP     HL
  03093 C9                RET

  03094 03                DEFB    03                ;--
  03095 01                DEFB    01
```

```
  03096 02              DEFB    02
  03097 04              DEFB    04
  03098 06              DEFB    06
  03099 08              DEFB    08
  0309A 09              DEFB    09
  0309B 0A              DEFB    0A
  0309C 05              DEFB    05

; Calculate new POS value (AF)
;
; I: --
; O: A = new POS value

  0309D E5              PUSH    HL              ;Save registers
  0309E D5              PUSH    DE
  0309F C5              PUSH    BC
  030A0 2A2040          LD      HL,(4020H)      ;HL = new cursor address
  030A3 E5              PUSH    HL              ;Save address
  030A4 CD6531          CALL    3165H           ;Calculate start of line
  030A7 EB              EX      DE,HL           ;DE -> start of line
  030A8 E1              POP     HL              ;HL = cursor address
  030A9 B7              OR      A               ;C-flag = 0
  030AA ED52            SBC     HL,DE           ;HL = cursor address - start
                                                ;of line = cursor position
                                                ;inside line
  030AC 7D              LD      A,L             ;A = POS value
  030AD C1              POP     BC              ;Restore registers
  030AE D1              POP     DE
  030AF E1              POP     HL
  030B0 C9              RET

; Test TAB value

  030B1 7B              LD      A,E             ;A = TAB value
  030B2 5F              LD      E,A             ;E = TAB value
  030B3 3A9C40          LD      A,(409CH)       ;A = output flag
  030B6 B7              OR      A               ;Test output flag
                                                ;Cassette output ?
  030B7 FA4A1E          JP      M,1E4AH         ;Yes: ?FC Error
                                                ;Screen output ?
  030BA 2805            JR      Z,30C1H         ;Yes: continue at 30C1H
                                                ;Printer output:
  030BC 3A9E40          LD      A,(409EH)       ;A = highest possible TAB
                                                ;value.
  030BF 1803            JR      30C4H           ;Continue at 30C4H

; Screen output

  030C1 3A9D40          LD      A,(409DH)       ;A = number of characters per
                                                ;line
  030C4 BB              CP      E               ;Desired TAB value bigger as
                                                ;length of line ?
  030C5 300B            JR      NC,30D2H        ;No: value ok
                                                ;Yes:
  030C7 C5              PUSH    BC              ;Save BC
  030C8 43              LD      B,E             ;Exchange A and E
  030C9 5F              LD      E,A
  030CA 78              LD      A,B
  030CB C1              POP     BC              ;Restore BC
  030CC 93              SUB     E               ;Subtract line length
  030CD 30FD            JR      NC,30CCH        ;Continue when > 0

  030CF 83              ADD     A,E             ;Undo last subtract
  030D0 5F              LD      E,A             ;E = new TAB value (never
                                                ;bigger then length of line)
  030D1 C9              RET

; TAB value is ok.

  030D2 7B              LD      A,E             ;Put value in A
  030D3 C9              RET
```

```
; SUB for PRINT@ (see 207AH)
;
; I: DE = @ argument

  030D4 21E703           LD      HL,03E7H          ;HL = 999
  030D7 DF               RST     18H               ;@ value > 999 ?
  030D8 E1               POP     HL                ;Restore PTP
  030D9 DA4A1E           JP      C,1E4AH           ;Yes: ?FC Error
                                                   ;No:
  030DC C37E20           JP      207EH             ;Done

; Calculate new POS value (as 309DH but with value in E and A)

  030DF CD9D30           CALL    309DH             ;Calculate POS
  030E2 5F               LD      E,A               ;Put result also in E
  030E3 C9               RET


; Screen routine (called via DCB call)
;
; I: IX -> Screen DCB (= 401DH)
;    C = character to be written

  030E4 DD6E03           LD      L,(IX+03H)        ;HL = actual cursor address
  030E7 DD6604           LD      H,(IX+04H)        ;(from DCB)
                                                   ;False DCB type ?
  030EA DA7D31           JP      C,317DH           ;Yes: continue at 317DH
                                                   ;No:
  030ED DD7E05           LD      A,(IX+05H)        ;Cursor is on ?
  030F0 B7               OR      A
  030F1 2005             JR      NZ,30F8H          ;Yes: continue at 30F8H
                                                   ;No:
  030F3 CD5930           CALL    3059H             ;Cursor off
  030F6 1803             JR      30FBH             ;Continue at 30F8H

  030F8 CD5F30           CALL    305FH             ;Cursor on
  030FB 79               LD      A,C               ;A = character to be output
  030FC FE20             CP      20H               ;Control character
  030FE 3822             JR      C,3122H           ;Yes: continue at 3122H
                                                   ;No:
  03100 FE80             CP      80H               ;Graphic character ?
  03102 D22230           JP      NC,3022H          ;Yes: continue at 3022H
                                                   ;No:
  03105 CD8431           CALL    3184H             ;Output character
  03108 7E               LD      A,(HL)            ;A = character at new cursor
                                                   ;position
  03109 57               LD      D,A               ;D = character
  0310A DD7E05           LD      A,(IX+05H)        ;Cursor is on ?
  0310D B7               OR      A
  0310E 2808             JR      Z,3118H           ;No: continue at 3118H
                                                   ;Yes:
  03110 DD7205           LD      (IX+05H),D        ;Store character
  03113 CD5F30           CALL    305FH             ;Cursor on
  03116 1803             JR      311BH             ;Continue at 311BH

  03118 CD5930           CALL    3059H             ;Cursor off
  0311B DD7503           LD      (IX+03H),L        ;Save new cursor address
  0311E DD7404           LD      (IX+04H),H
  03121 C9               RET


; Control character ( value < 32 ) recognized.
;
; I: A = control character
;    HL = cursor address

  03122 110831           LD      DE,3108H          ;Set return address
  03125 D5               PUSH    DE                ;to 3108H
```

```
  03126 FE08          CP      08H             ;Backspace ?
  03128 CADF31        JP      Z,31DFH         ;Yes: continue at 31DFH
                                              ;No:
  0312B FE0A          CP      0AH             ;Code < 10 ?
  0312D D8            RET     C               ;Yes: done

  0312E FE0E          CP      0EH             ;Cursor on ?
  03130 DAC931        JP      C,31C9H         ;Jump when character < 0EH

  03133 CAF831        JP      Z,31F8H         ;Yes: continue at 31F8H
                                              ;No:
  03136 FE0F          CP      0FH             ;Cursor off ?
  03138 CAFD31        JP      Z,31FDH         ;Yes: continue at 31FDH
                                              .No:
  0313B FE18          CP      18H             ;Character < 24
  0313D D8            RET     C               ;Yes: done
                                              ,No:
  0313E FE18          CP      18H             ;Cursor left ?
  03140 CAE531        JP      Z,31E5H         ;Yes: continue at 31E5H
                                              ;No:
  03143 FE19          CP      19H             ;Cursor right ?
  03145 CA3930        JP      Z,3039H         ;Yes: continue at 3039H
                                              ;No:
  03148 FE1A          CP      1AH             ;Cursor down ?
  0314A CA0032        JP      Z,3200H         ;Yes: continue at 3200H
                                              ;No:
  0314D FE1B          CP      1BH             ;Cursor up ?
  0314F CA1232        JP      Z,3212H         ;Yes: continue at 3212H
                                              ;No:
  03152 FE1C          CP      1CH             ;Cursor home ?
  03154 CAD431        JP      Z,31D4H         ;Yes: continue at 31D4H
                                              ;No:
  03157 FE1D          CP      1DH             ;Cursor to start of line ?
  03159 CAD931        JP      Z,31D9H         ;Yes: continue at 31D9H
                                              ;No:
  0315C FE1E          CP      1EH             ;CLear until end of line ?
  0315E 285F          JR      Z,31BFH         ;Yes: continue at 31BFH
                                              ;No:
  03160 FE1F          CP      1FH             ;Clear until end of screen ?
  03162 2845          JR      Z,31A9H         ;Yes: continue at 31A9H

  03164 C9            RET


; Calculate start of line address of current line
;
; I: HL = current cursor address
; O: HL = address of start of line

  03165 1100BC        LD      DE,0BC00H       ;DE = -4400H
  03168 0601          LD      B,01H           ;Counter = 1
  0316A 19            ADD     HL,DE           ;HL = current @ value
                                              ;(= address - 4400H)
  0316B 112800        LD      DE,0028H        ;DE = 40 (length of line)
  0316E B7            OR      A               ;C-flag = 0
  0316F ED52          SBC     HL,DE           ;Subtract 40
  03171 3803          JR      C,3176H         ;Jump when < 0

  03173 04            INC     B               ;Counter + 1
  03174 18F5          JR      316BH           ;Again

  03176 21D843        LD      HL,43D8H        ;HL = 4400H - 28H
  03179 19            ADD     HL,DE           ;Add length of 1 line
  0317A 10FD          DJNZ    3179H           ;Loop B times

  0317C C9            RET


; False DCB type detected

  0317D DD7E05        LD      A,(IX+05H)      ;Cursor is on ?
```

```
  03180 B7               OR      A
  03181 C0               RET     NZ               ;Yes: ok.
                                                  ;No:
  03182 7E               LD      A,(HL)           ;A = character at next
                                                  ;screen location
  03183 C9               RET


; Print character on screen
;
; I: HL -> screen location
;    A = character

  03184 CD7A30           CALL    307AH            ;Update character colour
  03187 77               LD      (HL),A           ;Put character in screen memory
  03188 23               INC     HL               ;Next screen location
  03189 11E847           LD      DE,47E8H         ;DE -> last screen location
                                                  ;on LGR screen + 1
  0318C DF               RST     18H              ;End of screen reached ?
  0318D D8               RET     C                ;No: done

  0318E 21E847           LD      HL,47E8H         ;HL = address of screen end
  03191 11D8FF           LD      DE,0FFD8H        ;DE = -40
  03194 19               ADD     HL,DE            ;HL = HL - 40 = start of last
                                                  ;line
  03195 E5               PUSH    HL               ;Save address
  03196 CD3936           CALL    3639H            ;Prepare registers

  03199 EDA0             LDI                      ;Move character memory
  0319B D9               EXX                      ;Switch registers
  0319C EDA0             LDI                      ;Move colour memory
  0319E D9               EXX                      ;Switch registers
  0319F 78               LD      A,B              ;Done ?
  031A0 B1               OR      C
  031A1 20F6             JR      NZ,3199H         ;No: move again

  031A3 D9               EXX                      ;Switch registers
  031A4 E1               POP     HL               ;Restore second registerset
  031A5 D1               POP     DE
  031A6 C1               POP     BC
  031A7 D9               EXX                      ;Switch registers
  031A8 E1               POP     HL               ;Restore start of last line

; Clear until end of screen (= CHR$(31) )

  031A9 11E847           LD      DE,47E8H         ;DE = highest LGR screen
                                                  ;address + 1
  031AC E5               PUSH    HL               ;Save cursor address

  031AD CDE031           CALL    31E0H            ;Clear character at cursor
                                                  ;address and set colour
  031B0 23               INC     HL               ;Next address
  031B1 DF               RST     18H              ;End reached ?
  031B2 20F9             JR      NZ,31ADH         ;No: clear next

  031B4 E1               POP     HL               ;Restore cursor address
  031B5 C9               RET


; RENUM statement
; ---------------

  031B6 FDE5             PUSH    IY               ;Save IY
  031B8 1868             JR      3222H            ;Continue at 3222H

; Return from RENUM

  031BA FDE1             POP     IY               ;Restore IY
  031BC C3832C           JP      2C83H            ;Renew all pointers in BASIC
                                                  ;program
```

```
; Clear until end of line (= CHR$(30) )

  031BF E5              PUSH    HL              ;Save cursor address
  031C0 CD6531          CALL    3165H           ;Calculate start address of
                                                ;current line
  031C3 EB              EX      DE,HL           ;result in DE, HL = 40
  031C4 19              ADD     HL,DE           ;HL = start address of next
                                                ;line
  031C5 EB              EX      DE,HL           ;Result in DE
  031C6 E1              POP     HL              ;Restore cursor address
  031C7 18E3            JR      31ACH           ;Continue at 31ACH

; Start new line (= CHR$(13) )

  031C9 CD1B36          CALL    361BH           ;Clear current line
  031CC 19              ADD     HL,DE           ;HL = start address next line
  031CD 11E847          LD      DE,47E8H        ;DE -> end of LGR screen + 1
  031D0 DF              RST     18H             ;End reached ?
  031D1 28BE            JR      Z,3191H         ;Yes: scroll
                                                ;No:
  031D3 C9              RET                     ;done

; Move cursor home (= CHR$(28) )

  031D4 210044          LD      HL,4400H        ;HL -> start of screen memory =
                                                ;new cursor address
  031D7 1803            JR      31DCH           ;Set as new address

; Move cursor to start of line (= CHR$(29) )

  031D9 CD6531          CALL    3165H           ;Calculate start of line
                                                ;address
  031DC C35F30          JP      305FH           ;Program CRTC with new values

; Backspace (= CHR$(8) )

  031DF 2B              DEC     HL              ;Cursor address - 1

  031E0 3E20            LD      A,20H           ;A = space character
  031E2 C3EE35          JP      35EEH           ;Continue at 35EEH

; Move cursor one position to the left (= CHR$(24) )

  031E5 E5              PUSH    HL              ;Save cursor address
  031E6 CD6531          CALL    3165H           ;Calculate start of line
                                                ;address
  031E9 EB              EX      DE,HL           ;Put result in DE
  031EA E1              POP     HL              ;Restore cursor address
  031EB DF              RST     18H             ;Are we at start of line ?
  031EC 2803            JR      Z,31F1H         ;Yes: add one line of
                                                ;characters for wrap around
  031EE 2B              DEC     HL              ;Cursor address - 1
  031EF 18EB            JR      31DCH           ;Set as new address

; Cursor to end of line

  031F1 2B              DEC     HL              ;HL - 1
  031F2 112800          LD      DE,0028H        ;DE = characters per line
  031F5 19              ADD     HL,DE           ;HL + 40
  031F6 18E4            JR      31DCH           ;Continue at 31DCH

; Cursor on

  031F8 7E              LD      A,(HL)          ;A = character at cursor pos.
  031F9 DD7705          LD      (IX+05H),A      ;Store it in DCB
  031FC C9              RET

; Cursor off

  031FD AF              XOR     A               ;A = 0
  031FE 18F9            JR      31F9H           ;Continue at 31F9H
```

```
; Move cursor one line down (= CHR$(26) )

  03200 CD0736          CALL    3607H           ;Set screen colour and DE = 40
  03203 19              ADD     HL,DE           ;Set cursor address into next
                                                ;line
  03204 B7              OR      A               ;Clear C-flag
  03205 3F              CCF                     ;C-flag = 1
  03206 11E847          LD      DE,47E8H        ;DE = highest LGR screen
                                                ;address + 1
  03209 DF              RST     18H             ;Cursor now beyound LGR screen ?
  0320A 3804            JR      C,3210H         ;No: continue at 3210H
                                                ;Yes:
  0320C 1140FC          LD      DE,0FC40H       ;DE = -960
  0320F 19              ADD     HL,DE           ;Add to cursor location to
                                                ;create wrap around to top line
  03210 18CA            JR      31DCH           ;Continue at 31DCH

; Move cursor one line up (= CHR$(27) )

  03212 CD0F36          CALL    360FH           ;Set screen colour and DE = -40
  03215 19              ADD     HL,DE           ;Set cursor address into
                                                ;previous line
  03216 110044          LD      DE,4400H        ;DE = lowest LGR screen address
  03219 DF              RST     18H             ;Cursor now before LGR screen ?
  0321A 3004            JR      NC,3220H        ;No: continue at 3220H
                                                ;Yes:
  0321C 11E803          LD      DE,03E8H        ;DE = 1000 (size of LGR screen)
  0321F 19              ADD     HL,DE           ;Add to cursor location to
                                                ;create a wrap around to bottom
                                                ;line
  03220 18BA            JR      31DCH           ;Continue at 31DCH


; RENUM entry from 31B8H
;
; I: HL = PTP

  03222 2B              DEC     HL              ;PTP - 1
  03223 D7              RST     10H             ;Character indicated
  03224 2006            JR      NZ,322CH        ;Yes: continue at 323FH

  03226 110A00          LD      DE,000AH        ;Starting LN = 10
  03229 D5              PUSH    DE              ;Save it
  0322A 1813            JR      323FH           ;Continue at 323FH

; Indicate starting line number and increment

  0322C D24A1E          JP      NC,1E4AH        ;?FC Error when not a number

  0322F CD5A1E          CALL    1E5AH           ;DE = starting LN
  03232 CF              RST     08H             ;Next character must be
  03233 2C              DEFB    ','             ;a comma
  03234 30F6            JR      NC,322CH        ;?FC Error when comma is not
                                                ;followed by a number
  03236 D5              PUSH    DE              ;Save starting LN
  03237 CD5A1E          CALL    1E5AH           ;DE = increment
  0323A 7A              LD      A,D             ;Increment = 0 ?
  0323B B3              OR      E
  0323C CA4A1E          JP      Z,1E4AH         ;Yes: ?FC Error

  0323F ED53E440        LD      (40E4H),DE      ;Store increment in system RAM
  03243 D1              POP     DE
  03244 ED53E240        LD      (40E2H),DE      ;Store starting LN in
                                                ;system RAM
  03248 FD2AF940        LD      IY,(40F9H)      ;IY -> End of program
  0324C 110001          LD      DE,0100H        ;DE = 256
  0324F FD19            ADD     IY,DE           ;IY = IY + 256
  03251 FDE5            PUSH    IY              ;Save end of program address
                                                ;+ 256
  03253 2AA440          LD      HL,(40A4H)      ;HL -> start of program
```

```
  03256 E5              PUSH    HL              ;Save it

; Build RENUM table

  03257 7E              LD      A,(HL)          ;End of program reached ?
  03258 23              INC     HL              ;Pointer + 1
  03259 B6              OR      (HL)            ;Next line pointer = 0 ?
  0325A 284A            JR      Z,32A6H         ;Yes: continue at 32A6H

  0325C 23              INC     HL              ;Pointer + 2
  0325D 23              INC     HL

  0325E CDBA33          CALL    33BAH           ;Search line for GOTO, GOSUB
                                                ;etc.
  03261 23              INC     HL              ;Pointer + 1
  03262 28F3            JR      Z,3257H         ;Process next line when no
                                                ;keyword was found
  03264 CDD833          CALL    33D8H           ;LN indicated? (in case of THEN
                                                ;or ELSE there can be either a
                                                ;command of a line number!)
  03267 2B              DEC     HL              ;Pointer - 1
  03268 28F4            JR      Z,325EH         ;No: continue search

  0326A 23              INC     HL              ;Pointer + 1
  0326B E5              PUSH    HL              ;Save program pointer
  0326C D5              PUSH    DE              ;Save table pointer
  0326D FDE5            PUSH    IY              ;Save end of program address
  0326F D1              POP     DE              ;and put it in DE
  03270 2AB140          LD      HL,(40B1H)      ;HL = TOPMEM
  03273 ED52            SBC     HL,DE           ;Subtract program end address
                                                ;Free memory space available ?
  03275 DA7A19          JP      C,197AH         ;No: ?OM Error
                                                ;Yes:
  03278 110400          LD      DE,0004H        ;Additional 4 bytes free ?
  0327B ED52            SBC     HL,DE
  0327D DA7A19          JP      C,197AH         ;No: ?OM Error
                                                ;Yes:
  03280 FD7000          LD      (IY+00H),B      ;Store number of digits of
                                                ;LN found
  03283 E1              POP     HL              ;Restore table pointer in HL
  03284 CD5A1E          CALL    1E5AH           ;DE = LN found
  03287 FD7301          LD      (IY+01H),E      ;Store LN
  0328A FD7202          LD      (IY+02H),D
  0328D FD360300        LD      (IY+03H),00H    ;And mark with 00H
  03291 CDB133          CALL    33B1H           ;Increase IY by 4
  03294 E1              POP     HL              ;Restore program pointer
  03295 2B              DEC     HL              ;Program pointer - 1

  03296 23              INC     HL              ;Program pointer + 1
  03297 7E              LD      A,(HL)          ;A = next character
  03298 FE20            CP      20H             ;Space found ?
  0329A 28FA            JR      Z,3296H         ;Yes: get next character

  0329C FE2C            CP      2CH             ;Comma found ? (with ON GOTO or
                                                ;ON GOSUB)
  0329E 2803            JR      Z,32A3H         ;Yes: continue ar 32A3H

  032A0 2B              DEC     HL              ;Program pointer - 1
  032A1 18BB            JR      325EH           ;Continue searching line

  032A3 23              INC     HL              ;Program pointer + 1
  032A4 18BE            JR      3264H           ;Serach next LN

; RENUM table completed
; Now use the new line numbers from the table

  032A6 FD3600FF        LD      (IY+00H),0FFH   ;Table end
  032AA E1              POP     HL              ;Restore program start
  032AB FDE1            POP     IY              ;Restore table start
  032AD ED5BE240        LD      DE,(40E2H)      ;DE = starting LN
  032B1 D5              PUSH    DE              ;Save new LN
```

```
  032B2 FDE5            PUSH    IY              ;Save table start
  032B4 E5              PUSH    HL              ;Save program start
  032B5 D5              PUSH    DE              ;Save starting LN
  032B6 CDC209          CALL    09C2H           ;BCDE = (HL): DE = pointer to
                                                ;next line, BC = line number
  032B9 7A              LD      A,D             ;Program end reached ?
  032BA B3              OR      E
  032BB 2841            JR      Z,32FEH         ;Yes: continue at 32FEH

  032BD EB              EX      DE,HL           ;HL -> next line
  032BE D1              POP     DE              ;Restore new LN
  032BF FDE5            PUSH    IY              ;Save table pointer

  032C1 FD7E00          LD      A,(IY+00H)      ;Table end reached ?
  032C4 3C              INC     A               ;(FFH + 1 = 00H)
  032C5 2821            JR      Z,32E8H         ;Yes: continue at 32E8H

  032C7 FD7E03          LD      A,(IY+03H)      ;A = table marker
  032CA B7              OR      A               ;Marker <> 0 ?
  032CB 2016            JR      NZ,32E3H        ;Yes: this entry has already
                                                ;the new line number
  032CD FD7E01          LD      A,(IY+01H)      ;No: Is the current line number
  032D0 B9              CP      C               ;already in the table ?
  032D1 2010            JR      NZ,32E3H

  032D3 FD7E02          LD      A,(IY+02H)
  032D6 B8              CP      B
  032D7 200A            JR      NZ,32E3H        ;No: check next entry
                                                ;Yes:
  032D9 FD7301          LD      (IY+01H),E      ;Insert new line number in
  032DC FD7202          LD      (IY+02H),D      ;the table
  032DF FD360301        LD      (IY+03H),01H    ;Mark entry
  032E3 CDB133          CALL    33B1H           ;Increase table pointer by 4
  032E6 18D9            JR      32C1H           ;Check next entry

; Table end reached, insert next line number

  032E8 FDE1            POP     IY              ;Restore table start
  032EA E5              PUSH    HL              ;Save program pointer
  032EB 2AE440          LD      HL,(40E4H)      ;HL = increment
  032EE 19              ADD     HL,DE           ;HL = new LN + increment =
                                                ;next new LN
                                                ;Next LN > 65535 ?
  032EF DA7A19          JP      C,197AH         ;Yes: ?OM Error (wrong error
                                                ;code !!)
  032F2 EB              EX      DE,HL           ;DE = new LN
  032F3 21F8FF          LD      HL,0FFF8H       ;New LN > 65529 ?
  032F6 ED52            SBC     HL,DE
  032F8 DA7A19          JP      C,197AH         ;Yes: ?OM Error (?!?)

  032FB E1              POP     HL              ;Restore program pointer
  032FC 18B7            JR      32B5H           ;Search for next line number in
                                                ;table
; Use new line numbers in program

  032FE D1              POP     DE              ;Correct stack
  032FF E1              POP     HL              ;Restore program start
  03300 FDE1            POP     IY              ;Restore table start
  03302 D1              POP     DE              ;Restore starting LN

  03303 7E              LD      A,(HL)          ;Program end reached ?
  03304 23              INC     HL
  03305 B6              OR      (HL)
  03306 CABA31          JP      Z,31BAH         ;Yes: RENUM done

  03309 23              INC     HL              ;Put new line number
  0330A 73              LD      (HL),E          ;into program
  0330B 23              INC     HL
  0330C 72              LD      (HL),D
  0330D CDBA33          CALL    33BAH           ;Search for GOTO, GOSUB etc.
  03310 23              INC     HL              ;Pointer + 1
```

```
  03311 2009            JR      NZ,331CH        ;Token found: continue at 33C1H

  03313 E5              PUSH    HL              ;Save pointer
  03314 2AE440          LD      HL,(40E4H)      ;HL = increment
  03317 19              ADD     HL,DE           ;Calculate next LN
  03318 EB              EX      DE,HL           ;DE = new LN
  03319 E1              POP     HL              ;Restore pointer
  0331A 18E7            JR      3303H           ;Put new LN into program

; GOTO, GOSUB etc. found

  0331C E5              PUSH    HL              ;Save pointer
  0331D D5              PUSH    DE              ;Save new LN
  0331E CDD833          CALL    33D8H           ;Transfer LN found into line
                                                ;buffer and determine the
                                                ;number of digits
  03321 D1              POP     DE              ;Restore new LN
  03322 E1              POP     HL              ;Restore pointer
  03323 2B              DEC     HL              ;Pointer - 1
  03324 28E7            JR      Z,330DH         ;Search for next token if no LN
                                                ;was found (e.g. after THEN
                                                ;or ELSE)
  03326 23              INC     HL              ;Pointer + 1
  03327 7E              LD      A,(HL)          ;A = next character
  03328 FE20            CP      20H             ;Space ?
  0332A 28FA            JR      Z,3326H         ;Yes: update pointer and take
                                                ;next character
  0332C D5              PUSH    DE              ;Save new LN
  0332D E5              PUSH    HL              ;Save pointer
  0332E FD6E01          LD      L,(IY+01H)      ;HL = new LN for this spot
  03331 FD6602          LD      H,(IY+02H)
  03334 CD9433          CALL    3394H           ;Put new LN in line buffer
                                                ;(in ASCII format!)
  03337 FD4E00          LD      C,(IY+00H)      ;C = number of digits (=length)
                                                ;of old LN
  0333A CDB133          CALL    33B1H           ;Increment IY to next entry
  0333D EB              EX      DE,HL           ;DE -> new LN in line buffer
  0333E E1              POP     HL              ;HL = program pointer
                                                ;(points to old LN)
  0333F CDF833          CALL    33F8H           ;Replace old LN at this spot
  03342 D1              POP     DE              ;Restore new LN
  03343 2B              DEC     HL              ;Pointer - 1

  03344 23              INC     HL              ;Pointer + 1
  03345 7E              LD      A,(HL)          ;Increment pointer until
  03346 FE20            CP      20H             ;the next non-space
  03348 28FA            JR      Z,3344H         ;character is found

  0334A FE2C            CP      2CH             ;Comma found ?
  0334C 2803            JR      Z,3351H         ;Yes: check for following LN
                                                ;No:
  0334E 2B              DEC     HL              ;Pointer - 1
  0334F 18BC            JR      330DH           ;Search for next token

; ',' found (with ON GOTO or ON GOSUB)

  03351 23              INC     HL              ;Pointer + 1
  03352 18C8            JR      331CH           ;Replace next LN


; Delete B characters from program
;
; I: B = number of characters to be deleted
;    HL -> program text
;    (from (HL) onwards, B bytes are removed from the program text)

  03354 D5              PUSH    DE              ;Save registers
  03355 C5              PUSH    BC
  03356 E5              PUSH    HL
  03357 E5              PUSH    HL
  03358 D1              POP     DE              ;DE -> program text
```

```
  03359 D5              PUSH    DE              ;Save DE twice
  0335A D5              PUSH    DE
  0335B 2AF940          LD      HL,(40F9H)      ;HL -> end of program
  0335E E5              PUSH    HL              ;Save it

  0335F 2B              DEC     HL              ;End address - 1
  03360 13              INC     DE              ;Program pointer + 1
  03361 10FC            DJNZ    335FH           ;Repeat until B = 0

  03363 22F940          LD      (40F9H),HL      ;Save new end address
  03366 E1              POP     HL              ;Restore old end address
  03367 C1              POP     BC              ;Restore program pointer
  03368 ED42            SBC     HL,BC           ;Calcute the number of bytes
  0336A 23              INC     HL              ;that have to be moved
  0336B E5              PUSH    HL              ;Put the resulting blocksize
  0336C C1              POP     BC              ;into BC
  0336D E1              POP     HL              ;Restore program pointer
  0336E EB              EX      DE,HL           ;DE -> characters to be deleted
                                                ;HL -> remaining program text
                                                ;(following the characters to
                                                ;be deleted)
  0336F EDB0            LDIR                    ;Move program text
  03371 E1              POP     HL              ;Restore registers
  03372 C1              POP     BC
  03373 D1              POP     DE
  03374 C9              RET


; Insert B characters into program
;
; I: B = number of characters to be inserted
;    HL -> program text
;    (from (HL) onwards, B bytes are inserted in the program text)

  03375 D5              PUSH    DE              ;Save registers
  03376 C5              PUSH    BC
  03377 E5              PUSH    HL
  03378 2AF940          LD      HL,(40F9H)      ;HL -> end of program
  0337B E5              PUSH    HL              ;Copy pointer
  0337C D1              POP     DE              ;into DE

  0337D 23              INC     HL              ;End address + 1
  0337E 10FD            DJNZ    337DH           ;Repeat until B = 0

  03380 22F940          LD      (40F9H),HL      ;Save new end address
  03383 C1              POP     BC              ;Restore program text pointer
  03384 C5              PUSH    BC              ;and save it again
  03385 E5              PUSH    HL              ;Save end address
  03386 B7              OR      A               ;C-flag = 0
  03387 ED42            SBC     HL,BC           ;Calculate the number of
                                                ;remaining characters upto
                                                ;program end
  03389 E5              PUSH    HL              ;Put the resulting blocksize
  0338A C1              POP     BC              ;into BC
  0338B 03              INC     BC              ;Adjust blocksize + 1
  0338C E1              POP     HL              ;Restore new end address
  0338D EB              EX      DE,HL           ;Put it in DE
  0338E EDB8            LDDR                    ;Move program text
  03390 E1              POP     HL
  03391 C1              POP     BC
  03392 D1              POP     DE
  03393 C9              RET


; Convert line number to ASCII format and put it in line buffer
;
; I: HL = line number
; O: B = number of digits (line number length)
;    HL -> line number in ASCII format

  03394 D5              PUSH    DE              ;Save DE
```

```
  03395 222141          LD      (4121H),HL      ;X = HL
  03398 010000          LD      BC,0000H        ;No formatting
  0339B 2AA740          LD      HL,(40A7H)      ;HL -> line buffer
  0339E E5              PUSH    HL              ;Save line buffer pointer
  0339F CD2F13          CALL    132FH           ;Decode number
  033A2 E1              POP     HL              ;Restore line buffer pointer
  033A3 0605            LD      B,05H           ;B = maximum length

  033A5 7E              LD      A,(HL)          ;A = character from line buffer
  033A6 D630            SUB     30H             ;Leading zero ?
  033A8 2005            JR      NZ,33AFH        ;No: done
                                                ;Yes:
  033AA 23              INC     HL              ;Line buffer pointer + 1
  033AB 10F8            DJNZ    33A5H           ;Get next character

  033AD 2B              DEC     HL              ;All digits = '0': pointer - 1
  033AE 04              INC     B               ;Counter + 1 (1 digit)

  033AF D1              POP     DE              ;Restore DE
  033B0 C9              RET


; Increment table pointer to next entry

  033B1 FD23            INC     IY
  033B3 FD23            INC     IY
  033B5 FD23            INC     IY
  033B7 FD23            INC     IY
  033B9 C9              RET


; Search line from (HL) for GOTO, GOSUB etc.
;
; I: HL -> line text
; O: A = token found
;    Z-flag = 1 if no token found
;    HL -> token found or end of line

  033BA 23              INC     HL              ;pointer + 1
  033BB 7E              LD      A,(HL)          ;A = char from program text
  033BC B7              OR      A               ;End of line ?
  033BD C8              RET     Z               ;Yes: return

  033BE FE8D            CP      8DH             ;Token for GOTO ?
  033C0 280C            JR      Z,33CEH         ;Yes: continue at 33CEH

  033C2 FE91            CP      91H             ;Token for GOSUB ?
  033C4 2808            JR      Z,33CEH

  033C6 FECA            CP      0CAH            ;Token for THEN ?
  033C8 2804            JR      Z,33CEH

  033CA FE95            CP      95H             ;Token for ELSE
  033CC 20EC            JR      NZ,33BAH        ;No: next character

; Token found

  033CE 2B              DEC     HL              ;Was it a Colour BASIC token ?
  033CF 7E              LD      A,(HL)          ;Does the previous byte
  033D0 FEFF            CP      0FFH            ;equal FFH ?
  033D2 23              INC     HL              ;Put pointer back
  033D3 7E              LD      A,(HL)          ;And character back

  033D4 28E4            JR      Z,33BAH         ;Yes: continue search

  033D6 A7              AND     A               ;No: Z-flag = 0
  033D7 C9              RET


; Determine length of line number found and put line number in line buffer
;
```

```
; I: HL -> line number (= token found + 1)
; O: B = number of digits (line number length)
;    DE -> line buffer (line number is in line buffer from (DE) onwards)
;    HL -> next character after the line number
;    Z-flag = 0 in case of digits found

  033D8 ED5BA740         LD      DE,(40A7H)       ;DE -> line buffer
  033DC D5               PUSH    DE               ;Save DE
  033DD 0600             LD      B,00H            ;Counter = 0

  033DF 7E               LD      A,(HL)           ;Get next character
  033E0 FE20             CP      20H              ;Space ?
  033E2 280B             JR      Z,33EFH          ;Yes: get next character

  033E4 FE30             CP      30H              ;Possible digit ?
  033E6 380A             JR      C,33F2H          ;No: done.

  033E8 FE3A             CP      3AH              ;Digit ?
  033EA 3006             JR      NC,33F2H         ;No: done.

  033EC 04               INC     B                ;Counter + 1
  033ED 12               LD      (DE),A           ;Store digit
  033EE 13               INC     DE               ;Buffer pointer + 1

  033EF 23               INC     HL               ;Line pointer + 1
  033F0 18ED             JR      33DFH            ;Loop

  033F2 AF               XOR     A                ;A = 00H
  033F3 12               LD      (DE),A           ;Mark line number end in line
                                                  ;buffer
  033F4 D1               POP     DE               ;Restore buffer address
  033F5 04               INC     B                ;Set Z-flag
  033F6 05               DEC     B                ;Z-flag = 0 when digits found
  033F7 C9               RET


; Shift program text in such a way, that a new line number can be inserted
; and then insert the new line number
;
; I: B = new line number length
;    C = old line number length
;    HL -> program text
;    (new line number is inserted from (HL) onwards)
;    DE -> new line number in ASCII format

  033F8 C5               PUSH    BC               ;Save lengths
  033F9 78               LD      A,B              ;A = new length
  033FA 99               SBC     A,C              ;- old length
  033FB 2810             JR      Z,340DH          ;Insert new line number when
                                                  ;both lengths are identical
  033FD 05               DEC     B                ;New length - 1
  033FE 2808             JR      Z,3408H          ;Old length > new length:
                                                  ;continue at 3408H
  03400 0D               DEC     C                ;Old length - 1
  03401 20F6             JR      NZ,33F9H         ;New length > old length:
                                                  ;continue at 33F9H
  03403 CD7533           CALL    3375H            ;Insert character
  03406 1805             JR      340DH            ;And transfer line number

; Old length is bigger than new length

  03408 41               LD      B,C              ;B = remaining length
  03409 05               DEC     B                ;B - 1
  0340A CD5433           CALL    3354H            ;Delete B characters
  0340D C1               POP     BC               ;Restore lengths

  0340E 1A               LD      A,(DE)           ;Transfer digit from line
  0340F 77               LD      (HL),A           ;buffer to program text
  03410 13               INC     DE               ;Line buffer pointer + 1
  03411 23               INC     HL               ;Program text pointer + 1
  03412 10FA             DJNZ    340EH            ;Repeat until B = 0
```

```
  03414 C9                RET


; Get key from keyboard and check for FKEY (see 3000H)

  03415 CD4900            CALL    0049H           ;Wait for key pressed
  03418 FE5C              CP      5CH             ;Possible FKEY ?
  0341A 383C              JR      C,3458H         ;No: continue at 3458H
                                                  ;Yes:
  0341C FE60              CP      60H             ;Is an FKEY ?
  0341E 3038              JR      NC,3458H        ;No: continue at 3458H

  03420 D65B              SUB     5BH             ;A = FKEY number 1..4


; Transfer FKEY text into line buffer
; A = FKEY code (01 to 08 for FKEY1 to FKEY8)

  03422 57                LD      D,A             ;D = FKEY code
  03423 78                LD      A,B             ;A = maximum number of
                                                  ;characters (see 05D9H)
  03424 FE07              CP      07H             ;Are 7 characters still
                                                  ;allowed ?
  03426 38ED              JR      C,3415H         ;No: ignore FKEY and get next
                                                  ;key input
  03428 7A                LD      A,D             ;FKEY code back into A
  03429 C5                PUSH    BC              ;Save counter
  0342A E5                PUSH    HL              ;Save buffer pointer

; Set HL as pointer to FKEY definition string belonging to FKEY pressed

  0342B 214943            LD      HL,4349H        ;HL -> FKEY definition strings
                                                  ;in system RAM - length of
                                                  ;one string
  0342E 110700            LD      DE,0007H        ;DE = length of FKEY string

  03431 19                ADD     HL,DE           ;HL -> next string
  03432 3D                DEC     A               ;Proper definition string ?
  03433 20FC              JR      NZ,3431H        ;No: next

  03435 D1                POP     DE              ;Restore buffer pointer
  03436 0607              LD      B,07H           ;B = length of FKEY string

  03438 7E                LD      A,(HL)          ;A = character of FKEY string
  03439 FE00              CP      00H             ;Is it a 00H (= execute a CR) ?
  0343B 2812              JR      Z,344FH         ;Yes, execute FKEY command

  0343D 12                LD      (DE),A          ;Store character
  0343E D5                PUSH    DE              ;Save buffer pointer
  0343F CD3300            CALL    0033H           ;Output byte
  03442 D1                POP     DE              ;Restore buffer pointer
  03443 23                INC     HL              ;Next position in FKEY string
  03444 13                INC     DE              ;Buffer pointer + 1
  03445 10F1              DJNZ    3438H           ;Process next

  03447 EB                EX      DE,HL           ;HL = buffer pointer
  03448 C1                POP     BC              ;Restore counter
  03449 78                LD      A,B             ;A = counter
  0344A D607              SUB     07H             ;Subtract 7 characters
  0344C 47                LD      B,A             ;Counter back in B
  0344D 18C6              JR      3415H           ;Wait for next key press

; Insert a 'RETURN' to execute FKEY command

  0344F EB                EX      DE,HL           ;HL = buffer pointer
  03450 78                LD      A,B             ;A = remaining length of FKEY
                                                  ;text
  03451 C1                POP     BC              ;Restore counter
  03452 80                ADD     A,B             ;A = maximum number of
                                                  ;characters + remaining length
```

```
  03453 D607           SUB     07H            ;Subtract 7 characters
  03455 47             LD      B,A            ;New counter back in B
  03456 3E0D           LD      A,0DH          ;A = 'RETURN'

; <SHIFT>+<FKEY> pressed ? (FKEY5 to FKEY8)

  03458 FE7C           CP      7CH            ;Possible FKEY pressed ?
  0345A DA0330         JP      C,3003H        ;No: continue at 3003H

  0345D FE80           CP      80H            ;FKEY pressed ?
  0345F D20330         JP      NC,3003H       ;No: continue at 3003H

  03462 D677           SUB     77H            ;A = FKEY code (05H to 08H)
  03464 18BC           JR      3422H          ;Continue at 3422H


; FKEY statement
; --------------

  03466 2B             DEC     HL             ;PTP - 1
  03467 D7             RST     10H            ;Get FKEY number
                                              ;Digit ?
  03468 D24A1E         JP      NC,1E4AH       ;No: ?FC Error

  0346B D630           SUB     30H            ;Convert character to number
  0346D F5             PUSH    AF             ;Save FKEY number
  0346E 23             INC     HL             ;PTP + 1
  0346F CF             RST     08H            ;Next character must be
  03470 D5             DEFB    0D5H           ;the token for '='
  03471 CF             RST     08H            ;Next character must be
  03472 22             DEFB    22H            ;double quotes
  03473 F1             POP     AF             ;A = FKEY number
  03474 FE01           CP      01             ;FKEY number < 1
  03476 DA4A1E         JP      C,1E4AH        ;Yes: ?FC Error

  03479 FE09           CP      09H            ;FKEY number > 8
  0347B 30EB           JR      NC,3468H       ;Yes: ?FC Error

  0347D E5             PUSH    HL             ;Save PTP
  0347E 214943         LD      HL,4349H       ;HL -> FKEY definitions - 7
  03481 110700         LD      DE,0007H       ;DE = 7 characters length
  03484 19             ADD     HL,DE          ;HL -> FKEY definition
  03485 3D             DEC     A              ;FKEY number - 1
                                              ;Down to 0 ?
  03486 20FC           JR      NZ,3484H       ;No: next FKEY text

  03488 EB             EX      DE,HL          ;DE -> proper FKEY definition
  03489 E1             POP     HL             ;HL = PTP
  0348A 2B             DEC     HL             ;PTP - 1

  0348B 0607           LD      B,07H          ;7 characters definition length

  0348D D7             RST     10H            ;Get next character
  0348E FE22           CP      22H            ;Is it double quotes ?
  03490 280C           JR      Z,349EH        ;Yes: end of new definition

  03492 FE00           CP      00H            ;Insert 'RETURN'
  03494 280A           JR      Z,34A0H        ;Yes: copy 00H and finish

  03496 12             LD      (DE),A         ;Copy into FKEY definition
  03497 13             INC     DE             ;Pointer + 1

  03498 10F3           DJNZ    348DH          ;Next character

  0349A 23             INC     HL             ;PTP + 1
  0349B CF             RST     08H            ;Next character must be
  0349C 22             DEFB    22H            ;double quotes
  0349D C9             RET

; End of text reached: fill remaining characters with spaces
```

```
  0349E 3E20             LD      A,' '            ;A = space

  034A0 12               LD      (DE),A           ;Copy into FKEY definition

  034A1 05               DEC     B                ;Next position in definition
                                                  ;All done ?
  034A2 2803             JR      Z,34A7H          ;Yes: continue ar 34A7H

  034A4 13               INC     DE               ;Pointer + 1
  034A5 18F9             JR      34A0H            ;Loop

  034A7 B7               OR      A                ;A = 00H ?
  034A8 C8               RET     Z                ;Yes: done

  034A9 18F0             JR      349BH            ;Test on double quotes


; Default definition for FKEY 1 to 8. (This table is copied into system RAM
; from 4350H onwards)

  034AB 4C495354202020  DEFB    'LIST   '        ;FKEY 1 = "LIST   "
  034B2 52554E20202020  DEFB    'RUN    '        ;FKEY 2 = "RUN    "
  034B9 4155544F202020  DEFB    'AUTO   '        ;FKEY 3 = "AUTO   "
  034C0 45444954202020  DEFB    'EDIT   '        ;FKEY 4 = "EDIT   "
  034C7 52454E554D2020  DEFB    'RENUM  '        ;FKEY 5 = "RENUM  "
  034CE 53595354454D00  DEFB    'SYSTEM',00H     ;FKEY 6 = "SYSTEM
  034D5 434C4F41442020  DEFB    'CLOAD  '        ;FKEY 7 = "CLOAD  "
  034DC 43534156452022  DEFB    'CSAVE "'        ;FKEY 8 = "CSAVE ""


; &H

  034E3 D7               RST     10H              ;Get character following &
  034E4 FE48             CP      48H              ;Is it 'H' (hexadecimal)
  034E6 203D             JR      NZ,3528H         ;No: test for &O

; Decode hexadecimal constant

  034E8 23               INC     HL               ;PTP + 1
  034E9 CD1635           CALL    34F3H            ;DE = constant
  034EC E5               PUSH    HL               ;Save PTP
  034ED EB               EX      DE,HL            ;HL = constant
  034EE CD9A0A           CALL    0A9AH            ;Write HL to X as INT
  034F1 E1               POP     HL               ;Restore PTP
  034F2 C9               RET

; DE = &H (HL)

  034F3 110000           LD      DE,0000H         ;Result = 0
  034F6 CD0135           CALL    3501H            ;Decode MSB
  034F9 57               LD      D,A              ;D = MSB
  034FB CD0135           CALL    3501H            ;Decode LSB
  034FE 5F               LD      E,A              ;E = LSB
  034FF 23               INC     HL               ;PTP + 1
  03500 C9               RET

; A = &H (HL)

  03501 CD0F35           CALL    350FH            ;Decode nibble
  03504 07               RLCA                     ;Shift bits 4-7
  03505 07               RLCA
  03506 07               RLCA
  03507 07               RLCA
  03508 23               INC     HL               ;PTP + 1
  03509 47               LD      B,A              ;Temporary result in B
  0350A CD0F35           CALL    350FH            ;Decode second nibble
  0350D 80               ADD     A,B              ;And add previous result
  0350E C9               RET

; Decode a hexadecimal digit
```

```
  0350F 7E              LD      A,(HL)          ;A = character
  03510 D630            SUB     '0'             ;Character < '0'
  03512 DA4A1E          JP      C,1E4AH         ;Yes: ?FC Error
  03515 FE0A            CP      10H             ;Character > '9'
  03517 D8              RET     C               ;No: value ok, done

  03518 D607            SUB     07H             ;Character < 'A'
  0351A DA4A1E          JP      C,1E4AH         ;Yes: ?FC Error
  0351D FE10            CP      10H             ;Character > 'F'
  0351F D24A1E          JP      NC,1E4AH        ;Yes: ?FC Error
  03522 C9              RET


; &O

  03523 FE4F            CP      4FH             ;Is it 'O' (octal)
  03525 C29719          JP      NZ,1997H        ;No: ?SN Error

; Decode octal constant

  03528 110000          LD      DE,0000H        ;Result = 0
  0352B 23              INC     HL              ;PTP + 1
  0352C CD4535          CALL    3545H           ;Decode highest octal digit
  0352F 07              RLCA                    ;Shift free bits 0-2
  03530 07              RLCA
  03531 07              RLCA
  03532 23              INC     HL              ;PTP + 1
  03533 47              LD      B,A             ;Temporary result in B
  03534 CD4535          CALL    3545H           ;Decode next octal digit
  03537 80              ADD     A,B             ;Add previous result
  03538 07              RLCA                    ;Shift free bits 0-2
  03539 07              RLCA
  0353A 07              RLCA
  0353B 23              INC     HL              ;PTP + 1
  0353C 47              LD      B,A             ;Temporary result in B
  0353D CD4535          CALL    3545H           ;Decode last octal digit
  03540 80              ADD     A,B             ;Add previous result
  03541 5F              LD      E,A             ;LSB value into E
  03542 23              INC     HL              ;PTP + 1
  03543 18A7            JR      34ECH           ;Write HL to X as INT

; Decode highest octal digit ('0' - '3')

  03545 7E              LD      A,(HL)          ;A = character
  03546 D630            SUB     '0'             ;Character < '0'
  03548 380D            JR      C,3557H         ;Yes: ?FC Error
  0354A FE04            CP      04H             ;Character < '4'
  0354C D8              RET     C               ;Yes: value ok, done
  0354D 1808            JR      3557H           ;No: ?FC Error

; Decode lower octal digit ('0' - '7')

  0354F 7E              LD      A,(HL)          ;A = character
  03550 D630            SUB     '0'             ;Character < '0'
  03552 3803            JR      C,3557H         ;Yes: ?FC Error
  03554 FE08            CP      08H             ;Character < '8'
  03556 D8              RET     C               ;Yes: value ok, done
  03557 C34A1E          JP      1E4AH           ;No: ?FC Error


; CALL statement
; --------------

  0355A CDFE34          CALL    34F3H           ;DE = address following CALL
  0355D E5              PUSH    HL              ;Save PTP
  0355E EB              EX      DE,HL           ;HL = address of routine
  0355F 116735          LD      DE,3567H        ;Set return address for routine
  03562 D5              PUSH    DE              ;that is called to 3567H
  03563 E9              JP      (HL)            ;activate routine

  03564 FF              RST     38H             ;--
```

```
  03565 FF              RST     38H
  03566 FF              RST     38H

; Return from CALL-routine

  03567 E1              POP     HL              ;Restore PTP
  03568 C9              RET


; Translation table for colour codes (is copied into system RAM from 4390H
; onwards). With the first Colour Genies, the COLOUR value was directly
; written into the colour-RAM (from F000H onwards).
; Due to hardware changes in later models, colour codes changed. In order to
; keep it compatible this table is used to regain the original colour settings.
; This table indicates for every COLOUR value, which value has to be
; written into the colour-RAM in order to regain the old order.
;
;                    Value in colour-RAM        COLOUR value

  03569 03              DEFB    03H             ;     1
  0356A 05              DEFB    05H             ;     2
  0356B 02              DEFB    02H             ;     3
  0356C 04              DEFB    04H             ;     4
  0356D 06              DEFB    06H             ;     5
  0356E 08              DEFB    08H             ;     6
  0356F 01              DEFB    01H             ;     7
  03560 0E              DEFB    0EH             ;     8
  03561 09              DEFB    09H             ;     9
  03572 10              DEFB    10H             ;    10
  03573 07              DEFB    07H             ;    11
  03574 0B              DEFB    0BH             ;    12
  03575 0C              DEFB    0CH             ;    13
  03576 0D              DEFB    0DH             ;    14
  03577 0A              DEFB    0AH             ;    15
  03578 0F              DEFB    0FH             ;    16


; Print text from (HL) onwards and generate sound (with error message)

  03579 CDA728          CALL    28A7H           ;Print text
  0357C 110900          LD      DE,0009H        ;Set various values to write
  0357F D5              PUSH    DE              ;into the PSG
  03580 110008          LD      DE,0800H
  03583 D5              PUSH    DE
  03584 113E10          LD      DE,103EH
  03587 D5              PUSH    DE
  03588 117800          LD      DE,0078H        ;DE = PSG value
  0358B 3E01            LD      A,01H           ;Write 0078H in frequency reg
  0358D CD2A3E          CALL    3E2AH           ;of channel A
  03590 D1              POP     DE
  03591 3E08            LD      A,08H           :Set amplitude of channel A to
                                                ;envelope shape use and
  03593 CD2A3E          CALL    3E2AH           ;switch on channel A
  03596 D1              POP     DE
  03597 3E0C            LD      A,0CH           ;Amplitude channel B = 8
  03599 CD2A3E          CALL    3E2AH           ;Amplitude channel C = 0
  0359C D1              POP     DE
  0359D 3E0D            LD      A,0DH
  0359F C3323E          JP      3E32H           ;Set envelope and return


; SUB for SHAPE (see 3CF2H).
; Load HL with the start address of the SHAPE table
;
; I:
; O: HL -> start of SHAPE table

  035A2 2AB140          LD      HL,(40B1H)      ;HL = TOPMEM
  035A5 23              INC     HL              ;HL -> start of SHAPE table
  035A6 C3F53C          JP      3CF5H           ;Continue at 3CF5H
```

```
; SUB for PRINT#

  035A9 CDB535          CALL    35B5H           ;Decode number
  035AC C33F02          JP      023FH           ;Write leader and sync


; SUB for INPUT#

  035AF CDB535          CALL    35B5H           ;Decode number
  035B2 C34C02          JP      024CH           ;Search for leader and sync


; Decode number at PRINT# and INPUT#

  035B5 AF              XOR     A               ;A = 0
  035B6 CD012B          CALL    2B01H           ;DE = number
  035B9 CF              RST     08H             ;Next character must be
  035BA 2C              DEFB    ','             ;a comma
  035BB 7B              LD      A,E             ;A = LSB
  035BC A2              AND     D               ;AND with MSB. MSB must be
                                                ;FFH because a negative number
                                                ;is required.
  035BD C602            ADD     A,02H           ;Number was -1 or -2 ?
  035BF D24A1E          JP      NC,1E4AH        ;No: ?FC Error

  035C2 C9              RET


; Set volume of indicated channel to 0 (not used)
; Is the value of the argument between 1 and 3, then the corresponding PSG
; channel volume will be set to 0. If no argument is given, the volume of
; all PSG channels will be set to 0.
; This routine can be used by redirecting e.g. the KILL command
; (so the command will be KILL n or KILL).
; POKE, &H4192,195 : POKE &H4193,53
;
; I: HL = PTP on argument or command/line end


  035C3 2B              DEC     HL              ;PTP -1
  035C4 D7              RST     10H             ;Channel number provided ?
  035C5 281A            JR      Z,35E1H         ;No: continue at 35E1H

  035C7 CD1C2B          CALL    2B1CH           ;A = Channel number
  035CA 0608            LD      B,08H           ;Register = 8
  035CC FE01            CP      01H             ;Channel 1 ?
  035CE 280B            JR      Z,35DBH         ;Yes: register number is ok

  035D0 04              INC     B               ;Register = 9
  035D1 FE02            CP      02H             ;Channel 2 ?
  035D3 2806            JR      Z,35DBH         ;Yes: register number is ok

  035D5 04              INC     B               ;Register = 10
  035D6 FE03            CP      03H             ;Channel 3 ?
  035D8 C24A1E          JP      NZ,1E4AH        ;No: ?FC Error

  035DB 78              LD      A,B             ;A = register number
  035DC 1E00            LD      E,00H           ;E = 00H
  035DE C3323E          JP      3E32H           ;Write to PSG (SOUND A,E)


; No channel number provided: set volume of all channels to 0.

  035E1 3E0A            LD      A,0AH           ;Switch off channel 2 and 3
  035E3 110000          LD      DE,0000H
  035E6 CD2A3E          CALL    3E2AH
  035E9 3E08            LD      A,08H           ;Switch off channel 1
  035EB C3323E          JP      3E32H
```

```
; SUB for screen output
; Write character in A at (HL) and update screen colour
;
; I: A = character
;    HL -> screen position

  035EE 77              LD      (HL),A          ;Write character
  035EF CD7A30          CALL    307AH           ;Update character colour
  035F2 23              INC     HL              ;Next position
  035F3 CD7A30          CALL    307AH           ;Update character colour
  035F6 2B              DEC     HL              ;Back to original position
  035F7 C9              RET


; SUB for screen output
; Update the colour of a screen location addressed by HL
;
; I: HL -> screen location of character in LGR screen memory

  035F8 1100AC          LD      DE,0AC00H       ;DE = difference between
                                                ;LGR screen memory (4400H)
                                                ;and corresponding colour
                                                ;memory (0F000H)
  035FB FE20            CP      ' '             ;Space character ?
  035FD C28030          JP      NZ,3080H        ;No: set colour according
                                                ;to current colour setting

; Spaces get the first colour in the colour code table

  03600 3A9043          LD      A,(4390H)       ;A = first entry in colour code
                                                ;table
  03603 19              ADD     HL,DE           ;Set pointer into colour mem
  03604 C38F30          JP      308FH           ;Set colour


; SUB for 'cursor one line down'

  03607 112800          LD      DE,0028H        ;DE = 40 (characters per line)
  0360A 3E20            LD      A,20H           ;Character = ' '
  0360C C37A30          JP      307AH           ;Output space (delete cursor)


; SUB for 'cursor one line up'

  0360F 11D8FF          LD      DE,0FFD8H       ;DE = -40
  03612 18F6            JR      360AH           ;Continue at 360AH


; SUB for 'cursor on/off'

  03614 2002            JR      NZ,3618H        ;Continue at 3618H (see 306EH)

  03616 3E20            LD      A,20H           ;--
  03618 C37A30          JP      307AH           ;Output character in A


; SUB for 'Carriage Return'

  0361B CD0A36          CALL    360AH           ;Delete cursor character
  0361E C36531          JP      3165H           ;Continue at 3165H


; SUB for COLOUR change (see 3017H)

  03621 322340          LD      (4023H),A       ;Save new COLOUR value
  03624 2A2040          LD      HL,(4020H)      ;HL = actual cursor address
  03627 C35F30          JP      305FH           ;Switch on cursor


; SUB for PRINT@ (see 2086H)
; Calculate new screen-POS
```

```
;
; I: DE = @ argument
; O: A = POS value

  0362A E5              PUSH    HL              ;Save HL
  0362B EB              EX      DE,HL           ;HL = @ argument
  0362C 112800          LD      DE,0028H        ;DE = 40 (length of 1 line)
  0362F 19              ADD     HL,DE           ;Add line length
  03630 B7              OR      A               ;C-flag = 0
  03631 ED52            SBC     HL,DE           ;Subtract line length until
  03633 30FC            JR      NC,3631H        ;result becomes negative

  03635 19              ADD     HL,DE           ;Compensate last subtract
  03636 7D              LD      A,L             ;A = L = actual POS
  03637 E1              POP     HL              ;Restore HL
  03638 C9              RET


; SUB for scroll (see 3196H)

  03639 D9              EXX                     ;Switch registers
  0363A F1              POP     AF              ;RET-address to AF
  0363B C5              PUSH    BC              ;Save registers
  0363C D5              PUSH    DE
  0363D E5              PUSH    HL
  0363E 1100F0          LD      DE,0F000H       ;Prepare second register set
  03641 2128F0          LD      HL,0F028H       ;for scroll of colour memory
  03644 01C003          LD      BC,03C0H
  03647 C5              PUSH    BC              ;Save BC as counter
  03648 D9              EXX                     ;Switch back registers
  03649 110044          LD      DE,4400H        ;Prepare first register set
  0364C 212844          LD      HL,4428H        ;for scroll of LGR screen mem.
  0364F C1              POP     BC              ;Restore counter
  03650 F5              PUSH    AF              ;RET address back on stack
  03651 C9              RET


; JOY statement
; -------------

  03652 7E              LD      A,(HL)          ;A = char. from program text
  03653 23              INC     HL              ;PTP + 1
  03654 FEDB            CP      0DBH            ;Token for INP ?
  03656 CAC536          JP      Z,36C5H         ;Yes: continue at 36C5H

  03659 FEA0            CP      0A0H            ;Token for OUT ?
  0365B CAB336          JP      Z,36B3H         ;Yes: continue at 36B3H

  0365E C39719          JP      1997H           ;?SN Error


; COLOUR function (right of = sign)
:
; *** WARNING: NEVER USE THIS! ***
; It can lead to a system crash or total program destruction
; If you want to perform this function do a PEEK(&H4023) + 1. Much safer....

  03661 3A2340          LD      A,(4023H)       ;A = colour value
  03664 3C              INC     A               ;Adjust to get proper value
                                                ;as with COLOUR n (1-16)
                                                ;Colour value zero ?
  03665 CA0D37          JP      Z,370DH         ;Yes: continue at 370DH

                                                ;Note: It would have been ok
                                                ;if the jump would have been
                                                ;unconditional! :-(

  03668 B8              CP      B               ;Compare with what ?
  03669 77              LD      (HL),A          ;Write A using PTP! Yikes!!!
```

```
; Read a BASIC program from I/O port (JOYINP)

  0366A 2AA440          LD      HL,(40A4H)      ;HL -> start of user BASIC RAM
  0366D 2B              DEC     HL              ;Pointer - 1

  0366E 0603            LD      B,03H           ;B = counter

  03670 E5              PUSH    HL              ;Save pointer
  03671 260F            LD      H,0FH           ;Select register 15 (port B)

  03673 CDBB3A          CALL    3ABBH           ;A = PSG port B
  03676 CB57            BIT     2,A             ;Strobe signal high ?
  03678 28F9            JR      Z,3673H         ;No, wait for strobe pulse

  0367A CDBB3A          CALL    3ABBH           ;A = contents of PSG register
  0367D CB57            BIT     2,A             ;Stobe signal low again ?
  0367F 20F9            JR      NZ,367AH        ;No, wait for strobe pulse end

  03681 260E            LD      H,0EH           ;Select register 14 (port A)
  03683 CDBB3A          CALL    3ABBH           ;A = PSG port A
  03686 E1              POP     HL              ;Restore pointer
  03687 77              LD      (HL),A          ;Save in memory
  03688 B7              OR      A               ;Set flags
  03689 23              INC     HL              ;Update memory pointer
                                                ;A = zero ?
  0368A 20E2            JR      NZ,366EH        ;No: next character of program
                                                ;3 consecutive zeros ?
  0368C 10E2            DJNZ    3670H           ;No: continue read

  0368E C9              RET


; Write a BASIC program to I/O port (JOYOUT)

  0368F 2AA440          LD      HL,(40A4H)      ;HL -> start of user BASIC RAM
  03692 2B              DEC     HL              ;Pointer - 1

  03693 ED5BF940        LD      DE,(40F9H)      ;DE -> end of program text + 2
  03697 7E              LD      A,(HL)          ;Get byte from program text
  03698 E5              PUSH    HL              ;Save pointer
  03699 6F              LD      L,A             ;Data in L
  0369A 260E            LD      H,0EH           ;Use PSG port A register
  0369C CDB23A          CALL    3AB2H           ;Output data by port A
  0369F 24              INC     H               ;Use PSG port B register
  036A0 3E04            LD      A,04H           ;Start strobe pulse
  036A2 CDB23A          CALL    3AB2H
  036A5 060A            LD      B,0AH           ;B = delay time
  036A7 10FE            DJNZ    36A7H           ;Wait

  036A9 AF              XOR     A               ;End strobe pulse
  036AA CDB23A          CALL    3AB2H           ;Output A
  036AD E1              POP     HL              ;Restore pointer
  036AE 23              INC     HL              ;Pointer + 1
  036AF DF              RST     18H             ;End of program reached ?
  036B0 20E5            JR      NZ,3697H        ;No, send next byte of program

  036B2 C9              RET


; JOYOUT
; ------

  036B3 E5              PUSH    HL              ;Save PTP
  036B4 2607            LD      H,07H           ;Select PSG enable register
  036B6 CDBB3A          CALL    3ABBH           ;A = contents of PSG register
  036B9 E63F            AND     3FH             ;Set port A and B to output
  036BB F6C0            OR      0C0H
  036BD CDB23A          CALL    3AB2H           ;Write to PSG enable register
  036C0 CD8F36          CALL    368FH           ;Write BASIC program
  036C3 E1              POP     HL              ;Restore PTP
```

```
  036C4 C9              RET


; JOYINP
; ------

  036C5 3E07            LD      A,07H           ;Select PSG enable register
  036C7 D3F8            OUT     (0F8H),A
  036C9 3E3F            LD      A,3FH           ;Set Port A and B to input
  036CB D3F9            OUT     (0F9H),A
  036CD CD6A36          CALL    366AH           ;Read BASIC program
  036D0 C3772C          JP      2C77H           ;Renew all pointers in program


; SWAP statement
; --------------

  036D3 CD0D26          CALL    260DH           ;Find or create 1st variable
  036D6 D5              PUSH    DE              ;Save pointer on 1st variable
  036D7 3AAF40          LD      A,(40AFH)       ;A = variable type of X
  036DA F5              PUSH    AF              ;Save it
  036DB CF              RST     08H             ;Next character must be
  036DC 2C              DEFB    ','             ;a comma
  036DD CD0D26          CALL    260DH           ;Find or create 2nd variable
  036E0 C1              POP     BC              ;Restore 1st variable type
  036E1 3AAF40          LD      A,(40AFH)       ;Take variable type of X
  036E4 B8              CP      B               ;Both variables same type ?
  036E5 C24A1E          JP      NZ,1E4AH        ;No: ?FC Error

; Now swap all bytes from both variables. The number of bytes to swap is
; indicated by the VT in B

  036E8 E3              EX      (SP),HL         ;HL -> 1st variable
  036E9 4E              LD      C,(HL)          ;C = byte from 1st var
  036EA 1A              LD      A,(DE)          ;A = byte from 2nd var
  036EB 77              LD      (HL),A          ;Byte from 2nd var in 1st var
  036EC 79              LD      A,C             ;A = byte from 1st var
  036ED 12              LD      (DE),A          ;Byte from 1st var in 2nd var
  036EE 23              INC     HL              ;Pointer on 1st var + 1
  036EF 13              INC     DE              ;Pointer on 2nd var + 1
  036F0 10F7            DJNZ    36E9H           ;Until all var bytes swapped

  036F2 E1              POP     HL              ;Restore PTP
  036F3 C9              RET


; X = SOUND(register)

  036F4 CF              RST     08H             ;Next character must be
  036F5 28              DEFB    '('             ;a '('
  036F6 CD1C2B          CALL    2B1CH           ;A = register parameter value
  036F9 F5              PUSH    AF              ;Save it
  036FA CF              RST     08H             ;Next character must be
  036FB 29              DEFB    ')'             ;a ')'
  036FC F1              POP     AF              ;Restore parameter value
  036FD FE10            CP      10H             ;In range (0 to 15)
  036FF D24A1E          JP      NC,1E4AH        ;No: ?FC Error

  03702 E5              PUSH    HL              ;Save PTP
  03703 67              LD      H,A             ;H = PSG register number
  03704 CDBB3A          CALL    3ABBH           ;A = contents PSG register H
  03707 C3723F          JP      3F72H           ;Put A in X and return


; X = SCALE

  0370A 3A1443          LD      A,(4314H)       ;A = current scale value
  0370D E5              PUSH    HL              ;Save PTP
  0370E C3723F          JP      3F72H           ;Put A in X as INT and return
```

```
; Table with colour codes (not used)

  03711 10                DEFB    10H
  03712 0D                DEFB    0DH
  03713 0E                DEFB    0EH
  03714 04                DEFB    04H
  03715 06                DEFB    06H
  03716 03                DEFB    03H
  03717 01                DEFB    01H
  03718 02                DEFB    02H
  03719 05                DEFB    05H
  0371A 07                DEFB    07H
  0371B 08                DEFB    08H
  0371C 09                DEFB    09H
  0371D 0A                DEFB    0AH
  0371E 0B                DEFB    0BH
  0371F 0C                DEFB    0CH
  03720 0F                DEFB    0FH

; Table with colour codes (not used)

  03721 10                DEFB    10H
  03722 0D                DEFB    0DH
  03723 06                DEFB    06H
  03724 04                DEFB    04H
  03725 0F                DEFB    0FH
  03726 03                DEFB    03H
  03727 09                DEFB    09H
  03728 02                DEFB    02H
  03729 01                DEFB    01H
  0372A 05                DEFB    05H
  0372B 07                DEFB    07H
  0372C 08                DEFB    08H
  0372D 0A                DEFB    0AH
  0372E 0B                DEFB    0BH
  0372F 0C                DEFB    0CH
  03730 0E                DEFB    0EH


: Table with colour codes (not used)

  03731 10                DEFB    10H
  03732 05                DEFB    05H
  03733 02                DEFB    02H
  03734 04                DEFB    04H
  03735 0E                DEFB    0EH
  03736 09                DEFB    09H
  03737 01                DEFB    01H
  03738 0A                DEFB    0AH
  03739 07                DEFB    07H
  0373A 06                DEFB    06H
  0373B 0D                DEFB    0DH
  0373C 03                DEFB    03H
  0373D 08                DEFB    08H
  0373E 0B                DEFB    0BH
  0373F 0C                DEFB    0CH
  03740 0F                DEFB    0FH


; Unused ROM space

  03741 FF                RST     38H               ;--
  03742 FF                RST     38H
  03743 FF                RST     38H
  03744 FF                RST     38H
  03745 FF                RST     38H
  03746 FF                RST     38H
  03747 FF                RST     38H
  03748 FF                RST     38H
  03749 FF                RST     38H
  0374A FF                RST     38H
```

```
0374B FF                RST     38H
0374C FF                RST     38H
0374D FF                RST     38H
0374E FF                RST     38H
0374F FF                RST     38H
03750 FF                RST     38H
03751 FF                RST     38H
03752 FF                RST     38H
03753 FF                RST     38H
03754 FF                RST     38H
03755 FF                RST     38H
03756 FF                RST     38H
03757 FF                RST     38H
03758 FF                RST     38H
03759 FF                RST     38H
0375A FF                RST     38H
0375B FF                RST     38H
0375C FF                RST     38H
0375D FF                RST     38H
0375E FF                RST     38H
0375F FF                RST     38H
03760 FF                RST     38H
03761 FF                RST     38H
03762 FF                RST     38H
03763 FF                RST     38H
03764 FF                RST     38H
03765 FF                RST     38H
03766 FF                RST     38H
03767 FF                RST     38H
03768 FF                RST     38H
03769 FF                RST     38H
0376A FF                RST     38H
0376B FF                RST     38H
0376C FF                RST     38H
0376D FF                RST     38H
0376E FF                RST     38H
0376F FF                RST     38H
03770 FF                RST     38H
03771 FF                RST     38H
03772 FF                RST     38H
03773 FF                RST     38H
03774 FF                RST     38H
03775 FF                RST     38H
03776 FF                RST     38H
03777 FF                RST     38H
03778 FF                RST     38H
03779 FF                RST     38H
0377A FF                RST     38H
0377B FF                RST     38H
0377C FF                RST     38H
0377D FF                RST     38H
0377E FF                RST     38H
0377F FF                RST     38H
03780 FF                RST     38H
03781 FF                RST     38H
03782 FF                RST     38H
03783 FF                RST     38H
03784 FF                RST     38H
03785 FF                RST     38H
03786 FF                RST     38H
03787 FF                RST     38H
03788 FF                RST     38H
03789 FF                RST     38H
0378A FF                RST     38H
0378B FF                RST     38H
0378C FF                RST     38H
0378D FF                RST     38H
0378E FF                RST     38H
0378F FF                RST     38H
03790 FF                RST     38H
03791 FF                RST     38H
```

```
  03792 FF              RST     38H
  03793 FF              RST     38H
  03794 FF              RST     38H
  03795 FF              RST     38H
  03796 FF              RST     38H
  03797 FF              RST     38H
  03798 FF              RST     38H
  03799 FF              RST     38H
  0379A FF              RST     38H
  0379B FF              RST     38H
  0379C FF              RST     38H
  0379D FF              RST     38H
  0379E FF              RST     38H
  0379F FF              RST     38H
  037A0 FF              RST     38H
  037A1 FF              RST     38H
  037A2 FF              RST     38H
  037A3 FF              RST     38H
  037A4 FF              RST     38H
  037A5 FF              RST     38H
  037A6 FF              RST     38H
  037A7 FF              RST     38H
  037A8 FF              RST     38H
  037A9 FF              RST     38H
  037AA FF              RST     38H
  037AB FF              RST     38H
  037AC FF              RST     38H
  037AD FF              RST     38H
  037AE FF              RST     38H
  037AF FF              RST     38H
  037B0 FF              RST     38H
  037B1 FF              RST     38H
  037B2 FF              RST     38H
  037B3 FF              RST     38H
  037B4 FF              RST     38H
  037B5 FF              RST     38H
  037B6 FF              RST     38H
  037B7 FF              RST     38H
  037B8 FF              RST     38H
  037B9 FF              RST     38H
  037BA FF              RST     38H
  037BB FF              RST     38H
  037BC FF              RST     38H
  037BD FF              RST     38H
  037BE FF              RST     38H
  037BF FF              RST     38H
  037C0 FF              RST     38H
  037C1 FF              RST     38H
  037C2 FF              RST     38H
  037C3 FF              RST     38H
  037C4 FF              RST     38H
  037C5 FF              RST     38H
  037C6 FF              RST     38H
  037C7 FF              RST     38H
  037C8 FF              RST     38H
  037C9 FF              RST     38H
  037CA FF              RST     38H


; Print text on screen (not used)
;
; I: HL -> text

  037CB D5              PUSH    DE              ;Save DE
  037CC 111D40          LD      DE,401DH        ;DE -> screen DCB
  037CF 1804            JR      37D5H           ;Continue at 36D5H

; Print text on printer (not used)
; The end of text must be indicated with 03H or 0DH.
; The 0DH is printed, the 03H not.
;
```

```
; I: HL -> text

  037D1 D5              PUSH    DE              ;Save DE
  037D2 112540          LD      DE,4025H        ;DE -> printer DCB

  037D5 E5              PUSH    HL              ;Save text pointer

  037D6 7E              LD      A,(HL)          ;A = character from text
  037D7 FE03            CP      03H             ;End of text ?
  037D9 2809            JR      Z,37E4H         ;Yes: continue at 37E4H

  037DB CD1B00          CALL    001BH           ;Print character (using DCB)
  037DE 7E              LD      A,(HL)          ;A = character from text
  037DF FE0D            CP      0DH             ;End of text ?
  037E1 23              INC     HL              ;Pointer + 1
  037E2 20F2            JR      NZ,37D6H        ;No: next character

  037E4 E1              POP     HL              ;Restore original text pointer
  037E5 D1              POP     DE              ;Restore DE
  037E6 C9              RET


; Conversion of the binary value in DE to 4 hex digits
; Used by DOS system call $HEXDE
; (reverse of the &H command)
;
; I: DE = binary value
;    HL -> memory space for storage of the 4 hex characters

  037E7 7A              LD      A,D             ;A = D
  037E8 CDEC37          CALL    37ECH           ;Convert A into 2 hex chars
  037EB 7B              LD      A,E             ;A = E

  037EC F5              PUSH    AF              ;Save binary value
  037ED 0F              RRCA                    ;Isolate upper nibble
  037EE 0F              RRCA
  037EF 0F              RRCA
  037F0 0F              RRCA
  037F1 CDF537          CALL    37F5H
  037F4 F1              POP     AF              ;Restore binary value
  037F5 E60F            AND     0FH             ;Isolate lower nibble
  037F7 C690            ADD     A,90H           ;Make hex character out of
  037F9 27              DAA                     ;the binary value 0-15
  037FA CE40            ADC     A,40H
  037FC 27              DAA
  037FD 77              LD      (HL),A          ;Save hex character
  037FE 23              INC     HL              ;Update pointer
  037FF C9              RET


; CTRC programming table for PAL standard
; The table entries are written into the CTRC register 15 downwards to
; register 0
; The table is copied into system RAM from 42F0H onwards

; LGR mode

  03800 01              DEFB    01H             ;Cursor position LSB
  03801 00              DEFB    00H             ;                MSB
  03802 00              DEFB    00H             ;Start address of page LSB
  03803 04              DEFB    04H             ;Start address of page MSB
  03804 07              DEFB    07H             ;Cursor stop scan line
  03805 C4              DEFB    C4H             ;Cursor start scan line
  03806 07              DEFB    07H             ;Scan lines/row
  03807 A0              DEFB    A0H             ;Interlace mode
  03808 1F              DEFB    1FH             ;VSYNC position
  03809 19              DEFB    19H             ;Character rows/frame
  0380A 00              DEFB    00H             ;VSYNC adjust
  0380B 26              DEFB    26H             ;Vertical total
  0380C 96              DEFB    96H             ;HSYNC width
  0380D 34              DEFB    34H             ;HSYNC position
```

```
  0380E 28              DEFB    28H             ;Characters/row
  0380F 46              DEFB    46H             ;Horizontal total

; FGR mode

  03810 00              DEFB    00H
  03811 00              DEFB    00H
  03812 00              DEFB    00H
  03813 08              DEFB    08H
  03814 00              DEFB    00H
  03815 20              DEFB    20H
  03816 01              DEFB    01H
  03817 20              DEFB    20H
  03818 74              DEFB    74H
  03819 66              DEFB    66H
  0381A 1F              DEFB    1FH
  0381B 7E              DEFB    7EH
  0381C 96              DEFB    96H
  0381D 34              DEFB    34H
  0381E 28              DEFB    28H
  0381F 46              DEFB    46H

; The next 3 bytes determine the baudrate at which characters are read/written
; from/to cassette tape. These values are copies into system RAM at
; 4310H, 4311H and 4312H. See also routines at 01FAH and 021FH.

  03820 46              DEFB    46H
  03821 4B              DEFB    4BH
  03822 69              DEFB    69H


; CTRC programming table for NTSC standard (not used)
; Using NTSC requires also hardware alterations on the video
; hardware. Therefor, this table is useless for non-NTSC
; computers

; LGR mode

  03823 01              DEFB    01H
  03824 00              DEFB    00H
  03825 00              DEFB    00H
  03826 04              DEFB    04H
  03827 07              DEFB    07H
  03828 C4              DEFB    C4H
  03829 07              DEFB    07H
  0382A A0              DEFB    A0H
  0382B 1B              DEFB    1BH
  0382C 19              DEFB    19H
  0382D 06              DEFB    06H
  0382E 1F              DEFB    1FH
  0382F 34              DEFB    34H
  03830 2E              DEFB    2EH
  03831 28              DEFB    28H
  03832 38              DEFB    38H

; FGR mode, NTSC standard

  03833 00              DEFB    00H
  03834 00              DEFB    00H
  03835 00              DEFB    00H
  03836 08              DEFB    08H
  03837 00              DEFB    00H
  03838 20              DEFB    20H
  03839 01              DEFB    01H
  0383A 20              DEFB    20H
  0383B 6E              DEFB    6EH
  0383C 66              DEFB    66H
  0383D 08              DEFB    08H
  0383E 7F              DEFB    7FH
  0383F 34              DEFB    34H
  03840 2E              DEFB    2EH
```

```
  03841 28              DEFB    28H
  03842 38              DEFB    38H

; Default values for timing of cassette I/O (not used)
; These values are used for the American version of the
; Colour Genie

  03843 4C              DEFB    4CH
  03844 51              DEFB    51H
  03845 71              DEFB    71H


; FCLS (without parameter)

  03846 3E00            LD      A,00H           ;A = FCOLOUR code (0)
  03848 1808            JR      3852H           ;Execute FCLS

; Continuation of FCLS routine of 3C87H (with parameter)

  0384A C4C23F          CALL    NZ,3FC2H        ;If there is a parameter:
                                                ;get parameter
  0384D FE04            CP      04H             ;Value in range (1 to 4)
  0384F D24A1E          JP      NC,1E4AH        ;No: ?FC Error


; SUB for FCLS (AF,BC,DE,HL)
; Perform an FCLS A
;
; I: A = FCOLOUR value

  03852 4F              LD      C,A             ;Value in C
                                                ;Build byte value for FGR
                                                ;memory to FCLS with the proper
                                                ;colour
  03853 0603            LD      B,03H           ;4 pixels per byte
  03855 07              RLCA                    ;Shift 2 positions
  03856 07              RLCA
  03857 B1              OR      C               ;or with C
  03858 10FB            DJNZ    3855H

  0385A 4F              LD      C,A             ;FCLS byte value in C
  0385B E5              PUSH    HL              ;Save PTP
  0385C 2AA440          LD      HL,(40A4H)      ;HL -> start of user BASIC RAM
  0385F 110148          LD      DE,4801H        ;DE -> start of FGR memory + 1
  03862 DF              RST     18H             ;FGR mode disabled at startup
                                                ;with <MOD SEL> ?
  03863 2809            JR      Z,386EH         ;Yes, nothing to do so return

  03865 210048          LD      HL,4800H        ;HL -> start of FGR memory
  03868 71              LD      (HL),C          ;Put FCLS value in first byte
  03869 01FF0F          LD      BC,0FFFH        ;BC = bytes to fill
  0386C EDB0            LDIR                    ;Fill all bytes in FGR memory

  0386E E1              POP     HL              ;Restore PTP
  0386F C9              RET


; This routine initializes the CRTC. The screen mode (LGR or FGR) is taken
; from the port 255 status byte

  03870 3A1C43          LD      A,(431CH)       ;Take port 255 status byte

; Entry with port 255 status byte already in A

  03873 E5              PUSH    HL              ;Save HL
  03874 21F042          LD      HL,42F0H        ;HL -> CRTC table for LGR mode
  03877 CB6F            BIT     5,A             ;LGR mode ?
  03879 2803            JR      Z,387EH         ;Yes: continue at 387EH

  0387B 210043          LD      HL,4300H        ;HL -> CRTC table for FGR mode
```

```
  0387E D3FF            OUT     (0FFH),A         ;Set CRTC mode
  03880 321C43          LD      (431CH),A        :Save port 255 output status

  03883 0610            LD      B,10H            ;B = CRTC registers 16
  03885 0EFA            LD      C,0FAH           ;C = CTRC register select port
  03887 05              DEC     B                ;register counter - 1
  03888 ED41            OUT     (C),B            ;Select register
  0388A 04              INC     B                ;Adjust B for OUTI
  0388B 0C              INC     C                ;C = CTRC data register port
  0388C EDA3            OUTI                     ;Write value from CRTC table
                                                 ;to register.
                                                 ;All registers done ?
  0388E 20F5            JR      NZ,3885H         ;No: update next register
  03890 E1              POP     HL               ;Restore HL
  03891 C9              RET


; This routine sets the screen to LGR and NBGRD. Then it prints the text
: addressed by HL
;
; I: HL -> text
; O: -

  03892 D9              EXX                      ;Save registers
  03893 3A1C43          LD      A,(431CH)        ;Take port 255 ouput status
  03896 E6DB            AND     0DBH             ;Set LGR and NBGRD. Leave other
                                                 ;bits as they are
  03898 CD7338          CALL    3873H            ;Initialize CRTC
  0389B D9              EXX                      ;Restore registers
  0389C CDA728          CALL    28A7H            ;Print text
  0389F C9              RET


; SUB for CONT (see 1DF2H)
; Program CRTC on last value and store last LN as actual LN


  038A0 D9              EXX                      ;Save registers
  038A1 CD7038          CALL    3870H            ;Initialize CRTC
  038A4 D9              EXX                      ;Restore registers
  038A5 22A240          LD      (40A2H),HL       ;Renew actual LN
  038A8 C9              RET


; FGR statement
; -------------

  038A9 3A1C43          LD      A,(431CH)        ;A = port 255 output status
  038AC CBEF            SET     5,A              ;FGR/LGR bit to FGR
  038AE 18C3            JR      3873H            ;Initialize CTRC


; LGR statement
; -------------

  038B0 3A1C43          LD      A,(431CH)        ;A = port 255 output status
  038B3 CBAF            RES     5,A              ;FGR/LGR bit to LGR
  038B5 18BC            JR      3873H            ;Initialize CRTC


; Old BRGD routine. Replaced by BGRD n at 3FE4H.

  038B7 0604            LD      B,04H            ;--
  038B9 1802            JR      38BDH


; NBGRD statement
; ---------------

  038BB 0600            LD      B,00H            ;BGRD value = 0
```

```
  038BD 3A1C43           LD      A,(431CH)       ;A = port 255 output status
  038C0 E6FB             AND     0FBH            ;Clear old BGRD bits
  038C2 B0               OR      B               ;Set new BGRD bits
  038C3 321C43           LD      (431CH),A       ;Store status in system RAM
  038C6 D3FF             OUT     (0FFH),A        ;Set new BGRD
  038C8 C9               RET


; COLOUR statement
; ----------------

  038C9 CDC23F           CALL    3FC2H           ;A = colour value - 1
  038CC FE10             CP      10H             ;Value in range (1 to 16) ?
  038CE D24A1E           JP      NC,1E4AH        ;No: ?FC Error

  038D1 322340           LD      (4023H),A       ;Store new value in system RAM
  038D4 C9               RET


; FCOLOUR statement
; -----------------
; The FCOLOUR token is stored in 3 bytes! (FFH, 81H, 52H)
; Because of an error in the old ROM, only the keyword FCOLOU was recognized.
; The missing 'R' had to be stored seperately.

  038D5 CF               RST     08H             ;Next character must be a 'R'
  038D6 52               DEFB    'R'             ;becuase token is only FCOLOU
  038D7 CDC23F           CALL    3FC2H           ;A = fcolour value - 1
  038DA FE04             CP      04H             ;Value in range (1 to 4) ?
  038DC 30F0             JR      NC,38CEH        ;No: ?FC Error

  038DE 321343           LD      (4313H),A       ;Store new FCOLOUR value
  038E1 C9               RET


; SUB for tokenizing (see 1C15H)
; Scan Colour-keywords table when the end of the normal keyword table
; has been reached.
;
; I: A = current table character
;    B = token counter
;    C = current text character
;    DE = text pointer
;    HL = table pointer

  038E2 E67F             AND     7FH             ;Table end reached ?
  038E4 C0               RET     NZ              ;No: return

  038E5 EB               EX      DE,HL           ;HL = text pointer
  038E6 112F39           LD      DE,392FH        ;DE -> Colour keyword table
  038E9 C5               PUSH    BC              ;Save token counter
  038EA 067F             LD      B,7FH           ;Set new token counter
  038EC 7E               LD      A,(HL)          ;A = current text character
  038ED FE61             CP      61H             ;Possible lower case ?
  038EF 3806             JR      C,38F7H         ;No: ok, continue at 38F7H

  038F1 FE7B             CP      7BH             ;Lower case ?
  038F3 3002             JR      NC,38F7H        ;No: ok, continue at 38F7H

  038F5 E65F             AND     5FH             ;Convert to upper case
  038F7 4E               LD      C,(HL)          ;C = current text character
  038F8 EB               EX      DE,HL           ;HL = table pointer
  038F9 23               INC     HL              ;Table pointer + 1
  038FA B6               OR      (HL)            ;Next keyword reached ?
  038FB F2F938           JP      P,38F9H         ;No: increment pointer until
                                                 ;next keyword reached
  038FE 04               INC     B               ;Token counter + 1
  038FF 7E               LD      A,(HL)          ;A = keyword table character
  03900 E67F             AND     7FH             ;End of keyword table reached ?
  03902 2829             JR      Z,392DH         ;Yes: return
```

```
  03904 B9              CP      C               ;Compare with text character
                                                ;The same ?
  03905 20F2            JR      NZ,38F9H        ;No: try next keyword

  03907 EB              EX      DE,HL           ;DE = keyword table pointer
  03908 E5              PUSH    HL              ;Save text pointer
  03909 13              INC     DE              ;Table pointer + 1
  0390A 1A              LD      A,(DE)          ;A = next table character
  0390B B7              OR      A               ;Next keyword reached ?
  0390C FA1E39          JP      M,391EH         ;Yes: all characters match!
                                                ;continue ar 391EH
  0390F 4F              LD      C,A             ;No: C = table character
  03910 23              INC     HL              ;Text pointer + 1
  03911 7E              LD      A,(HL)          ;A = next text character
  03912 FE61            CP      61H             ;Convert to upper case
  03914 3802            JR      C,3918H

  03916 E65F            AND     5FH
  03918 B9              CP      C               ;Compare to table character
  03919 28EE            JR      Z,3909H         ;Next character when indentical

  0391B E1              POP     HL              ;Restore old text pointer
  0391C 18D9            JR      38F7H           ;And compare next keyword

; Keyword found

  0391E F1              POP     AF              ;Text pointer
  0391F F1              POP     AF              ;Old token counter
  03920 F1              POP     AF              ;Remove RET address to 1C18H
  03921 D1              POP     DE              ;and to 1C3DH
  03922 D1              POP     DE              ;Restore counter of characters
                                                ;per line
  03923 E3              EX      (SP),HL         ;Save text pointer, restore
                                                ;buffer pointer
  03924 36FF            LD      (HL),0FFH       ;Store FFH in buffer as
                                                ;Colour-keyword identifier
  03926 78              LD      A,B             ;A = token value of keyword
                                                ;that has been found
  03927 42              LD      B,D             ;BC = DE
  03928 4B              LD      C,E
  03929 D1              POP     DE              ;Restore text pointer in DE
  0392A C3573D          JP      3D57H           ;Store token

; End of keyword table reached, no keyword matched

  0392D C1              POP     BC              ;Restore old counters
  0392E F1              POP     AF              ;Remove RET address to 1C18H
  0392F C9              RET                     ;Return to 1C3DH


; Keyword table for Colour BASIC statements

  03930 C34F4C4F5552    DEFB    80H+'C','OLOUR'         ;COLOUR
  03936 C6434F4C4F55    DEFB    80H+'F','COLOU'         ;FCOLOUR
  0393C CB4559504144    DEFB    80H+'K','EYPAD'         ;KEYPAD
  03942 CA4F59          DEFB    80H+'J','OY'            ;JOY
  03945 D04C4F54        DEFB    80H+'P','LOT'           ;PLOT
  03949 C64752          DEFB    80H+'F','GR'            ;FGR
  0394C CC4752          DEFB    80H+'L','GR'            ;LGR
  0394F C6434C53        DEFB    80H+'F','CLS'           ;FCLS
  03953 D04C4159        DEFB    80H+'P','LAY'           ;PLAY
  03957 C34952434C45    DEFB    80H+'C','IRCLE'         ;CIRCLE
  0395D D343414C45      DEFB    80H+'S ,'CALE'          ;SCALE
  03962 D348415045      DEFB    80H+'S','HAPE'          ;SHAPE
  03967 CE5348415045    DEFB    80H+'N','SHAPE'         ;NSHAPE
  0396D D85348415045    DEFB    80H+'X','SHAPE'         ;XSHAPE
  03973 D041494E54      DEFB    80H+'P','AINT'          ;PAINT
  03978 C3504F494E54    DEFB    80H+'C','POINT'         ;CPOINT
  0397E CE504C4F54      DEFB    80H+'N','PLOT'          ;NPLOT
  03983 D34F554E44      DEFB    80H+'S','OUND'          ;SOUND
  03988 C3484152        DEFB    80H+'C','HAR'           ;CHAR
```

```
  0398C D2454E554D      DEFB    80H+'R','ENUM'          ;RENUM
  03991 D3574150        DEFB    80H+'S','WAP'           ;SWAP
  03995 C64B4559        DEFB    80H+'F','KEY'           ;FKEY
  03999 C3414C4C        DEFB    80H+'C','ALL'           ;CALL
  0399D D64552494659    DEFB    80H+'V','ERIFY'         ;VERIFY
  039A3 C2475244        DEFB    80H+'B','GRD'           ;BGRD
  039A7 CE42475244      DEFB    80H+'N','BGRD'          ;NBGRD
  039AC 80              DEFB    80H                     ;End of table marker


; SUB for LIST (see 2BA9H)
; Establish start of keyword table.
;
; O: HL -> keyword table

  039AD FE80            CP      80H             ;Token value = FFH? (FFH - 7FH)
  039AF 215016          LD      HL,1650H        ;HL -> BASIC keyword table
  039B2 C0              RET     NZ              ;No: HL is set properly: return

  039B3 E1              POP     HL              ;RET address to HL
  039B4 E3              EX      (SP),HL         ;RET address back,
                                                ;Line pointer to HL
  039B5 7E              LD      A,(HL)          ;A = next character
  039B6 D67F            SUB     7FH             ;Is it a token ?
  039B8 5F              LD      E,A             ;E = character - 7FH
  039B9 23              INC     HL              ;Pointer + 1
  039BA E3              EX      (SP),HL         ;Save pointer,
                                                ;RET address in HL
  039BB E5              PUSH    HL              ;Save RET address
  039BC 2A8C43          LD      HL,(438CH)      ;HL -> Colour keyword table
  039BF C9              RET


; SUB for program loop (see 1D67H)
; Establish start address of jump address table

  039C0 FE7F            CP      7FH             ;Colour token found ?
  039C2 2808            JR      Z,39CCH         ;Yes: continue at 39CCH

  039C4 FE3C            CP      3CH             ;Command found ?
  039C6 D2E72A          JP      NC,2AE7H        ;No: continue at 2AE7H

  039C9 C36A1D          JP      1D6AH           ;Execute command

; Colour token found

  039CC 23              INC     HL              ;PTP + 1
  039CD 7E              LD      A,(HL)          ;Get real token value
  039CE D680            SUB     80H             ;Subtract 80H
  039D0 07              RLCA                    ;*2 = table offset
  039D1 4F              LD      C,A             ;BC = table offset
  039D2 0600            LD      B,00H
  039D4 EB              EX      DE,HL           ;DE = PTP
  039D5 2A8E43          LD      HL,(438EH)      ;HL  -> Colour jump table
  039D8 C3721D          JP      1D72H           ;Get jump address as execute
                                                ;routine

; Address table for Colour BASIC statements

  039DB C938            DEFW    38C9H           ;COLOUR
  039DD D538            DEFW    38D5H           ;FCOLOUR
  039DF 9719            DEFW    1997H           ;KEYPAD  (= ?SN Error!)
  039E1 5236            DEFW    3652H           ;JOY
  039E3 C13B            DEFW    3BC1H           ;PLOT
  039E5 A938            DEFW    38A9H           ;FGR
  039E7 B038            DEFW    38B0H           ;LGR
  039E9 833C            DEFW    3C83H           ;FCLS
  039EB 613D            DEFW    3D61H           ;PLAY
  039ED F83A            DEFW    3AF8H           ;CIRCLE
  039EF F13A            DEFW    3AF1H           ;SCALE
  039F1 DD3C            DEFW    3CDDH           ;SHAPE
```

```
  039F3 D83C            DEFW    3CD8H           ;NSHAPE
  039F5 D33C            DEFW    3CD3H           ;XSHAPE
  039F7 383E            DEFW    3E38H           ;PAINT
  039F9 9719            DEFW    1997H           ;CPOINT  (= ?SN Error!)
  039FB BE3B            DEFW    3BBEH           ;NPLOT
  039FD 953F            DEFW    3F95H           ;SOUND
  039FF A83F            DEFW    3FA8H           ;CHAR
  03A01 B631            DEFW    31B6H           ;RENUM
  03A03 D336            DEFW    36D3H           ;SWAP
  03A05 6634            DEFW    3466H           ;FKEY
  03A07 5A35            DEFW    355AH           ;CALL
  03A09 4935            DEFW    3F33H           ;VERIFY
  03A0B E43F            DEFW    3FE4H           ;BGRD
  03A0D BB38            DEFW    38BBH           ;NBGRD


; X = KEYPAD1, KEYPAD2, or KEYPAD(n)

  03A0F 7E              LD      A,(HL)          ;A = char from program text
  03A10 23              INC     HL              ;PTP + 1
  03A11 E5              PUSH    HL              ;Save PTP
  03A12 FE31            CP      '1'             ;Is it a '1' ? (KEYPAD1)
  03A14 CAD83A          JP      Z,3AD8H         ;Yes: continue at 3AD8H

  03A17 FE32            CP      '2'             ;Is it a '2' ? (KEYPAD2)
  03A19 CADC3A          JP      Z,3ADCH         ;Yes: continue at 3ADCH

  03A1C C3C33A          JP      3AC3H           ;Check if (n) set


; X = JOY1X, JOY1Y, JOY2X, JOY2Y or JOY(n)

  03A1F 7E              LD      A,(HL)          ;A = character at PTP
  03A20 FE28            CP      '('             ;Is it a '('
  03A22 2826            JR      Z,3A4AH         ;Yes: continue at 3A4AH

  03A24 1604            LD      D,04H
  03A26 FE32            CP      '2'             ;Is it a '2' ? (JOY2)
  03A28 2807            JR      Z,3A31H         ;Yes: continue at 3A31H

  03A2A CB3A            SRL     D               ;D = D / 2
  03A2C FE31            CP      '1'             ;Is it a '1' ? (JOY1)
  03A2E C29719          JP      NZ,1997H        ;No: ?SN Error

  03A31 D7              RST     10H             ;Get next non-space character
  03A32 FE59            CP      'Y'             ;Char = 'Y' ? (JOY1Y,JOY2Y)
  03A34 2805            JR      Z,3A3BH         ;Yes: continue at 3A3BH

  03A36 15              DEC     D               ;D = D - 1
  03A37 FE58            CP      'X'             ;Char = 'X' ? (JOY1X,JOY2X)
  03A39 20F3            JR      NZ,3A2EH        ;No: ?SN Error

                                                ;JOY1X: D = 1   JOY1Y: D = 2
                                                ;JOY2X: D = 3   JOY2Y: D = 4

  03A3B 23              INC     HL              ;PTP + 1
  03A3C E5              PUSH    HL              ;Save PTP
  03A3D CD5E3A          CALL    3A5EH           ;Get joystick value
  03A40 E63F            AND     3FH             ;Set to value between  0-63
  03A42 3C              INC     A               ;Adjust to 1 to 64
  03A43 1816            JR      3A5BH           ;Put A in X as INT and return


; Not used

  03A45 18F4            JR      3A3BH           ;Get joystick value
  03A47 00              NOP                     ;--
  03A48 00              NOP
  03A49 00              NOP
```

```
; JOY (n)

  03A4A 23              INC     HL              ;PTP + 1
  03A4B CD1C2B          CALL    2B1CH           ;Get parameter
  03A4E FE08            CP      08H             ;In range (0-7)
  03A50 D24A1E          JP      NC,1E4AH        ;No: ?FC Error

  03A53 57              LD      D,A             ;Parameter in D
  03A54 CF              RST     08H             ;Next character must be
  03A55 29              DEFB    ')'             ;a ')'
  03A56 14              INC     D               ;D + 1
  03A57 E5              PUSH    HL              ;Save PTP
  03A58 CD5E3A          CALL    3A5EH           ;Get joystick value
  03A5B C3723F          JP      3F72H           ;Put A in X as INT and return


; Get joystick value
; I: D = parameter with joystick ID (range: 1 to 8)
;       D = 1: JOY1X / D = 2: JOY1Y / D = 3: JOY2X / D = 4: JOY2Y
;       The remaining values (5 to 8) can be used for a second pair
;       of joysticks (or 4 other analogue inputs)
; O: Value return by joystick

  03A5E CDA93A          CALL    3AA9H           ;Set port A to output,
                                                ;port B to input
  03A61 AF              XOR     A               ;A = 0;
  03A62 1E80            LD      E,80H           ;E,7 = 1

  03A64 B3              OR      E               ;Set bit in A
  03A65 6F              LD      L,A             ;L = A
  03A66 260E            LD      H,0EH           ;Select PSG port A
  03A68 CDB33A          CALL    3AB3H           ;Write A to port A
  03A6B 24              INC     H               ;Select PSG Port B
  03A6C CDBB3A          CALL    3ABBH           ;A = contents of port B
  03A6F D5              PUSH    DE              ;Save counter D

  03A70 17              RLA                     ;Shift required bit into C-flag
  03A71 15              DEC     D               ;Required bit in C-flag ?
  03A72 20FC            JR      NZ,3A70H        ;No: shift again

  03A74 D1              POP     DE              ;Restore counter
  03A75 7D              LD      A,L             ;Last output value in A
                                                ;Required bit = '1' ?
  03A76 3803            JR      C,3A7BH         ;Yes: continue at 3A7BH

  03A78 7B              LD      A,E             ;A = bitmask
  03A79 2F              CPL                     ;Invert it
  03A7A A5              AND     L               ;And with L
  03A7B CB3B            SRL     E               ;Shift E 1 bit to the right
                                                ;All 8 bits done ?
  03A7D 30E5            JR      NC,3A64H        ;No: loop

  03A7F C9              RET


; Old KEYPAD routine (now at 3A0FH)

  03A80 C3143F          JP      3F14H           ;Was: KEYPAD1
  03A83 00              NOP                     ;--
  03A84 C3183F          JP      3F18H           ;Was: KEYPAD2

; KEYPAD routine
; Read value from keypad and store in A
;
; I: D = keypad bit mask (FEH for KEYPAD1, F7H for KEYPAD2)
; O: A = value from keypad

  03A87 CDA93A          CALL    3AA9H           ;Set port A to output,
                                                ;port B to input
  03A8A 1EE4            LD      E,0E4H          ;E = LSB of start address
                                                ;of KEYPAD table
```

```
  03A8C 0603            LD      B,03H           ;Test 2 columns

  03A8E 6A              LD      L,D             ;L = keypad address
  03A8F 260E            LD      H,0EH           ;Select port A
  03A91 CDB33A          CALL    3AB3H           ;Select keypad
  03A94 24              INC     H               ;Next PSG register
  03A95 CDBB3A          CALL    3ABBH           ;Get keypad value
  03A98 0E04            LD      C,04H           ;Test 4 rows
  03A9A 1F              RRA                     ;Shift next bit into C-flag
                                                ;Key pressed ?
  03A9B 3008            JR      NC,3AA5H        ;Yes: continue at 3AA5H

  03A9D 1C              INC     E               ;Table offset + 1
  03A9E 0D              DEC     C               ;Row counter - 1
  03A9F 20F9            JR      NZ,3A9AH        ;Check next row

  03AA1 CB02            RLC     D               ;Address next column
  03AA3 10E9            DJNZ    3A8EH           ;Read next column

  03AA5 163A            LD      D,3AH           ;DE -> keypad table at proper
                                                ;offset
  03AA7 1A              LD      A,(DE)          ;A = keypad value
  03AA8 C9              RET

; Prepare port A for output and port B for input

  03AA9 2607            LD      H,07H           ;Select PSG enable register
  03AAB CDBB3A          CALL    3ABBH           ;A = contents of PSG register
  03AAE E63F            AND     3FH             ;Set port A to output (bit 7)
  03AB0 F640            OR      40H             ;and port B to input (bit 6)

; Write A into PSG register addressed by H
;
; I: H = PSG register number
;    A = data to write into register

  03AB2 6F              LD      L,A             ;L = A

  03AB3 0EF8            LD      C,0F8H
  03AB5 ED61            OUT     (C),H           ;Select PSG register
  03AB7 0C              INC     C
  03AB8 ED69            OUT     (C),L           ;Write new contents in PSG
  03ABA C9              RET


; Read PSG register addressed by H
;
; I: H = PSG register number
; O: A = contents of data register

  03ABB 0EF8            LD      C,0F8H          ;C = PSG select register
  03ABD ED61            OUT     (C),H           ;Select register H
  03ABF 0C              INC     C               ;C = PSG data register
  03AC0 ED78            IN      A,(C)           ;A = data from data register
  03AC2 C9              RET

; KEYPAD(n)

  03AC3 E1              POP     HL              ;Restore PTP
  03AC4 2B              DEC     HL              ;Adjust PTP
  03AC5 CF              RST     08H             ;Next character must be
  03AC6 28              DEFB    '('             ;a '('
  03AC7 CDC23F          CALL    3FC2H           ;Get n - 1
  03ACA FE02            CP      02H             ;In range (1 or 2)
  03ACC D24A1E          JP      NC,1E4AH        ;No: ?FC Error
  03ACF F5              PUSH    AF              ;save n
  03AD0 CF              RST     08H             ;Next character must be
  03AD1 29              DEFB    ')'             ;a ')'
  03AD2 F1              POP     AF              ;Restore parameter
  03AD3 E5              PUSH    HL              ;Save PTP
  03AD4 FE01            CP      01H             ;n = 1 ?
```

```
  03AD6 2804              JR      Z,3ADCH         ;Yes: continue at 3ADCH

; KEYPAD1

  03AD8 16FE              LD      D,0FEH          ;Set bitmask for keypad 1
  03ADA 1802              JR      3ADEH           ;Continue at 3ADEH

; KEYPAD2

  03ADC 16F7              LD      D,0F7H          ;Set bitmask for keypad 2

  03ADE CD873A            CALL    3A87H           ;A = value from keypad
  03AE1 C3723F            JP      3F72H           ;Put A in X and return


; Keypad table with the return values for the 12 keys on the keypad

  03AE4 03                DEFB    03H             ;'3'
  03AE5 06                DEFB    06H             ;'6'
  03AE6 09                DEFB    09H             ;'9'
  03AE7 0C                DEFB    0CH             ;Fire button (right)
  03AE8 02                DEFB    02H             ;'2'
  03AE9 05                DEFB    05H             ;'5'
  03AEA 08                DEFB    08H             ;'8'
  03AEB 0A                DEFB    0AH             ;'0'
  03AEC 01                DEFB    01H             ;'1'
  03AED 04                DEFB    04H             ;'4'
  03AEE 07                DEFB    07H             ;'7'
  03AEF 0B                DEFB    0BH             ;Fire button (left)
  03AF0 00                DEFB    00H             ;No key pressed


; SCALE statement
; ---------------

  03AF1 CD1C2B            CALL    2B1CH           ;Get new scale value
  03AF4 321443            LD      (4314H),A       ;and save it in system RAM
  03AF7 C9                RET


; CIRCLE statement
; ----------------

  03AF8 CD1C2B            CALL    2B1CH           ;Get X-coordinate
  03AFB F5                PUSH    AF              ;Save X-coordinate
  03AFC CF                RST     08H             ;Next character must be
  03AFD 2C                DEFB    ','             ;a comma
  03AFE CD1C2B            CALL    2B1CH           ;Get Y-coordinate
  03B01 F5                PUSH    AF              ;Save Y-coordinate
  03B02 CF                RST     08H             ;Next character must be
  03B03 2C                DEFB    ','             ;a comma
  03B04 CD1C2B            CALL    2B1CH           ;Get radius
  03B07 D1                POP     DE              ;D = Y-coordinate
  03B08 C1                POP     BC              ;B = X-coordinate
  03B09 4A                LD      C,D             ;C = Y-coordinate

; B = X-coordinate, C = Y-coordinate, A = radius

  03B0A E5                PUSH    HL              ;Save PTP
  03B0B 57                LD      D,A             ;Radius in D = Y-distance
  03B0C 1E00              LD      E,00H           ;X-distance = 0
  03B0E 2680              LD      H,80H           ;H = step value

; Plot next 8 points
                                                  ;Y-distance < X-distance
                                                  ;(circle is closed) ?
  03B10 FA523B            JP      M,3B52H         ;Yes: restore PTP and return

  03B13 CD7A3B            CALL    3B7AH           ;Plot 4 points and swap X-
                                                  ;and Y-distance
  03B16 CD7A3B            CALL    3B7AH           ;Plot 4 points and swap back
```

```
  03B19 CD5E3B           CALL    3B5EH          ;Negate distances
  03B1C C5               PUSH    BC             ;Save registers
  03B1D D5               PUSH    DE
  03B1E E5               PUSH    HL
  03B1F 2E80             LD      L,80H          ;Bit 7 of L = 1: This causes
                                                ;an overflow with the first
                                                ;ADD HL,HL to bit 0 of H
  03B21 63               LD      H,E            ;H = X-distance
  03B22 0608             LD      B,08H          ;Process 8 bits
  03B24 1E00             LD      E,00H          ;E = 0, D = Y-distance

; Calculate next step for Y distance

  03B26 29               ADD     HL,HL          ;Shift next bit from H
                                                ;to C-flag
  03B27 ED52             SBC     HL,DE          ;HL = HL - DE - C-flag
  03B29 3803             JR      C,3B2EH        ;Jump when underflow. Because
                                                ;E = 0, this can only happen
                                                ;when H < D.
  03B2B 23               INC     HL             ;HL + 1 (= L + 1)
  03B2C 1801             JR      3B2FH          ;Loop

  03B2E 19               ADD     HL,DE          ;Reverse subtract
  03B2F 10F5             DJNZ    3B26H          ;Loop

  03B31 7D               LD      A,L            ;A = L
  03B32 E1               POP     HL             ;Restore registers
  03B33 D1               POP     DE
  03B34 C1               POP     BC
  03B35 84               ADD     A,H            ;A = A + step value
                                                ;C-flag = 1 if A + H > 255
  03B36 67               LD      H,A            ;H = new step value
  03B37 7A               LD      A,D            ;A = Y-distance
  03B38 2E00             LD      L,00H          ;L = 0;
  03B3A 9D               SBC     A,L            ;A = Y-distance - C-flag
  03B3B 57               LD      D,A            ;D = new Y-distance
  03B3C 1C               INC     E              ;X-distance + 1
  03B3D 7A               LD      A,D            ;A = Y-distance
  03B3E BB               CP      E              ;X-distance > Y-distance ?
  03B3F C3103B           JP      3B10H          ;Continue at 3B10H

; PLOT B + E , C + D

  03B42 C5               PUSH    BC             ;Save registers
  03B43 D5               PUSH    DE
  03B44 E5               PUSH    HL
  03B45 7B               LD      A,E            ;A = X-distance
  03B46 80               ADD     A,B            ;+ centre-coordinates
  03B47 6F               LD      L,A            ;Results in X-coordinate
  03B48 7A               LD      A,D            ;A = Y-distance
  03B49 81               ADD     A,C            ;+ centre-coordinates
  03B4A 67               LD      H,A            ;Results in Y-coordinate
  03B4B CD8A3B           CALL    3B8AH          ;PLOT L , H
  03B4E E1               POP     HL             ;Restore registers
  03B4F D1               POP     DE
  03B50 C1               POP     BC
  03B51 C9               RET

; End CIRCLE

  03B52 E1               POP     HL             ;Restore PTP
  03B53 C9               RET

; Negate X-distance (E = -E)

  03B54 7B               LD      A,E            ;A = E
  03B55 ED44             NEG                    ;A = -A
  03B57 5F               LD      E,A            ;new value back in E
  03B58 C9               RET

; Negate Y-distance (D = -D)
```

```
  03B59 7A              LD      A,D             ;A = D
  03B5A ED44            NEG                     ;A = -A
  03B5C 57              LD      D,A             ;new value back in D
  03B5D C9              RET

; Negate X- and Y-distance

  03B5E CD543B          CALL    3B54H           ;Negate X-distance
  03B61 CD593B          CALL    3B59H           ;Negate Y-distance
  03B64 C9              RET

; PLOT B - E , C + D

  03B65 CD543B          CALL    3B54H           ;Negate X-distance
  03B68 CD423B          CALL    3B42H           ;PLOT B + E , C + D
  03B6B C9              RET

; PLOT B + E , C - D

  03B6C CD593B          CALL    3B59H           ;Negate Y-distance
  03B6F CD423B          CALL    3B42H           ;PLOT B + E , C + D
  03B72 C9              RET

; Swap X- and Y-distance

  03B73 7B              LD      A,E             ;L = E
  03B74 6F              LD      L,A
  03B75 7A              LD      A,D             ;E = D
  03B76 5F              LD      E,A
  03B77 7D              LD      A,L             ;D = L
  03B78 57              LD      D,A
  03B79 C9              RET

; Plot 4 points and exchange X- and Y-distance

  03B7A CD423B          CALL    3B42H           ;PLOT B + E, C + D
                                                ;(bottom right)
  03B7D CD653B          CALL    3B65H           ;PLOT B - E, C + D
                                                ;(bottom left)
  03B80 CD6C3B          CALL    3B6CH           ;PLOT B - E, C - D
                                                ;(top left)
  03B83 CD653B          CALL    3B65H           ;PLOT B + E, C - D
                                                ;(top right)
  03B86 CD733B          CALL    3B73H           ;Exchange distances
                                                ;for second call, see 3B13H)
  03B89 C9              RET


; SUB for PLOT, PAINT, SHAPE and CIRCLE
; PLOT L , H (AF,BC,DE,HL)
;
; I: L = X-coordinate
;    H = Y-coordinate
;    (4313H) = FCOLOUR value

  03B8A 3A1343          LD      A,(4313H)       ;A = FCOLOUR value
  03B8D E603            AND     03H             ;Clear unused bits

; PLOT L , H
; as 3B8AH but A = FCOLOUR value

  03B8F 4F              LD      C,A             ;C = FCOLOUR value
  03B90 3E9F            LD      A,9FH           ;A = X-max. (159)
  03B92 BD              CP      L               ;Beyond X-max ?
  03B93 D8              RET     C               ;Yes: done

  03B94 3E65            LD      A,65H           ;A = Y-max (101)
  03B96 BC              CP      H               ;Beyond Y-max ?
  03B97 D8              RET     C               ;Yes: done
```

```
  03B98 7D              LD      A,L              ;A = X-coordinate
  03B99 6C              LD      L,H              ;L = Y-coordinate
  03B9A 2600            LD      H,00H            ;HL = Y-coordinate
  03B9C 54              LD      D,H              ;DE = HL
  03B9D 5D              LD      E,L
  03B9E 29              ADD     HL,HL            ;HL = HL * 40
  03B9F 29              ADD     HL,HL            ;(one row in FGR mode
  03BA0 19              ADD     HL,DE            ;corresponds with 40 bytes
  03BA1 29              ADD     HL,HL            ;in FGR screen memory)
  03BA2 29              ADD     HL,HL
  03BA3 29              ADD     HL,HL
  03BA4 5F              LD      E,A              ;E = X-coordinate
  03BA5 CB3B            SRL     E                ;E = E * 4
  03BA7 CB3B            SRL     E                ;(one byte in FGR memory
                                                 ;contains 4 pixels)
  03BA9 1648            LD      D,48H            ;DE = start of FGR memory +
                                                 ;X-coordinate * 4
  03BAB 19              ADD     HL,DE            ;Add Y-coordinate * 40 to get
                                                 ;the byte in FGR screen memory
                                                 ;that has to be used
  03BAC E603            AND     03H              ;Compute position of pixel
  03BAE 3C              INC     A                ;inside byte
  03BAF 47              LD      B,A              ;B = X-coordinate MOD 4
  03BB0 3EFC            LD      A,0FCH           ;A = 11111100 (2 bits/pixel)
  03BB2 0F              RRCA                     ;Shift the 2 zero bits to the
  03BB3 0F              RRCA                     ;required position
  03BB4 CB09            RRC     C                ;Shift FCOLOUR code along
  03BB6 CB09            RRC     C
  03BB8 10F8            DJNZ    3BB2H            ;Repeat until counter = 0

  03BBA A6              AND     (HL)             ;Get byte of required pixel
  03BBB B1              OR      C                ;and set new colour
  03BBC 77              LD      (HL),A           ;Save new byte value
  03BBD C9              RET


; NPLOT statement
; ---------------

  03BBE 0600            LD      B,00H            ;Set NPLOT
  03BC0 3A0603          LD      A,(0306H)        ;--


; PLOT statement
; --------------

* 03BC1   0603          LD      B,03H            ;Set PLOT
  03BC3 3A1343          LD      A,(4313H)        ;A = FCOLOUR value
  03BC6 F5              PUSH    AF               ;Save it on stack
  03BC7 A0              AND     B                ;And it with B to get either
                                                 ;black (NPLOT) or the current
                                                 ;value (PLOT)
  03BC8 321343          LD      (4313H),A        ;Save new FCOLOUR to use
  03BCB CD7B3C          CALL    3C7BH            ;'TO' token indicated ?
  03BCE 382F            JR      C,3BFFH          ;No: continue at 3BFFH

  03BD0 3A1543          LD      A,(4315H)        ;Get last X-value and
  03BD3 F5              PUSH    AF               ;save it on stack
  03BD4 3A1643          LD      A,(4316H)        ;Get last Y-value and
  03BD7 F5              PUSH    AF               ;save it on stack
  03BD8 CD1C2B          CALL    2B1CH            ;Get next X-value and
  03BDB 321543          LD      (4315H),A        ;save it in system RAM
  03BDE F5              PUSH    AF               ;and on stack
  03BDF CF              RST     08H              ;Next character must be
  03BE0 2C              DEFB    ','              ;a comma
  03BE1 CD1C2B          CALL    2B1CH            ;Get next Y-value and
  03BE4 321643          LD      (4316H),A        ;save it in system RAM
  03BE7 D9              EXX                      ;switch register set
  03BE8 6F              LD      L,A              ;L' = Y2
  03BE9 D1              POP     DE               ;Restore X-value
  03BEA 62              LD      H,D              ;H' = X2
```

```
  03BEB C1              POP     BC              ;Restore last Y-value
  03BEC D1              POP     DE              ;Restore last X-value
  03BED 58              LD      E,B             ;E' = Y1, D' = X1
  03BEE D9              EXX                     ;Switch register set
  03BEF E5              PUSH    HL              ;Save PTP
  03BF0 D9              EXX                     ;Switch register set
  03BF1 CD1F3C          CALL    3C1FH           ;PLOT  D , E  TO  H , L
  03BF4 E1              POP     HL              ;Restore PTP
  03BF5 CD7B3C          CALL    3C7BH           ;'TO' indicated ?
  03BF8 28D6            JR      Z,3BD0H         ;Yes, get next X,Y-coordinates

  03BFA F1              POP     AF              ;Restore old FCOLOUR value
  03BFB 321343          LD      (4313H),A       ;and store it in system RAM
  03BFE C9              RET

; Get new X,Y-coordinates ('TO' not at the start)

  03BFF CD1C2B          CALL    2B1CH           ;Get X-coordinate
  03C02 F5              PUSH    AF              ;Save X-coordinate
  03C03 CF              RST     08H             ;Next character must be
  03C04 2C              DEFB    ','             ;a comma
  03C05 CD1C2B          CALL    2B1CH           ;Get Y-coordinate
  03C08 F5              PUSH    AF              ;Save Y-coordinate
  03C09 CD7B3C          CALL    3C7BH           ;'TO' indicated ?
  03C0C 30CA            JR      NC,3BD8H        ;Yes: get second pair of
                                                ;X,Y-coordinates
  03C0E F1              POP     AF              ;No: Restore Y-coordinate
  03C0F 321643          LD      (4316H),A       ;Save it in system RAM
  03C12 57              LD      D,A             ;D = Y-coordinate
  03C13 F1              POP     AF              ;Restore X-coordinate
  03C14 321543          LD      (4315H),A       ;Save it in system RAM
  03C17 5F              LD      E,A             ;E = X-coordinate
  03C18 EB              EX      DE,HL           ;DE = PTP, H = Y-, L = X-coor.
  03C19 D5              PUSH    DE              ;Save PTP
  03C1A CD8A3B          CALL    3B8AH           ;PLOT  L , H
  03C1D 18D5            JR      3BF4H           ;Restore PTP and previous
                                                ;FCOLOUR value

; SUB for PLOT
; PLOT  D , E  TO  H , L  (AF,BC,DE,HL)
; I: D = X-Coordinate of starting point
;    E = Y-Coordinate of starting point
;    H = X-Coordinate of ending point
;    L = Y-Coordinate of ending point

  03C1F CDC63C          CALL    3CC6H           ;PLOT  H , L (plot end point)
  03C22 DF              RST     18H             ;HL = DE ? (start and end are
                                                ;identical)
  03C23 C8              RET     Z               ;Yes: done.

  03C24 00              NOP                     ;--
  03C25 00              NOP
  03C26 D5              PUSH    DE              ;Save X1,Y1
  03C27 7B              LD      A,E             ;A = Y1
  03C28 95              SUB     L               ;A = Y1 - Y2
                                                ;Negative result ?
  03C29 DC8B3C          CALL    C,3C8BH         ;Yes: negate result and set
                                                ;C-flag = 1
  03C2C CB19            RR      C               ;Put C-flag in bit 7 of C
  03C2E CB39            SRL     C               ;Shift C to the right
  03C30 47              LD      B,A             ;B = Y-diff (always positive!)
  03C31 CB39            SRL     C               ;Shift C to the right
  03C33 CB39            SRL     C               ;Shift C to the right
  03C35 7A              LD      A,D             ;A = X1
  03C36 94              SUB     H               ;A = X1 - X2
                                                ;Negative result ?
  03C37 DC8B3C          CALL    C,3C8BH         ;Yes: negate result and set
                                                ;C-flag = 1
  03C3A CB19            RR      C               ;Put C-flag in bit 7 of C
  03C3C 37              SCF                     ;C-flag = 1
  03C3D CB19            RR      C               ;Shift C to the right, bit 7=1
```

```
  03C3F B8                CP      B                ;X-diff < Y-diff ?
  03C40 384D              JR      C,3C8FH          ;Yes: swap lower and upper
                                                   ;nibble of C, E = X-diff,
                                                   ;D = Y-diff
  03C42 57                LD      D,A              ;No: D = X-diff
  03C43 78                LD      A,B
  03C44 5F                LD      E,A              ;E = Y-diff

; D now contains the bigger difference and E contains the smaller difference.
; Both nibbles of C indicate the angle; the upper nibble for D
; and the lower nibble for E

  03C45 C5                PUSH    BC               ;Save angle
  03C46 E5                PUSH    HL               ;Save X2,Y2
  03C47 7A                LD      A,D              ;A = bigger difference
  03C48 4F                LD      C,A              ;C = A
  03C49 3E00              LD      A,00H            ;A = 0
  03C4B 57                LD      D,A              ;D = 0
  03C4C 47                LD      B,A              ;B = 0, so BC = bigger diff.
  03C4D 67                LD      H,A              ;H = 0
  03C4E 7B                LD      A,E              ;A = smaller difference
  03C4F 6F                LD      L,A              ;L = A, so HL = smaller diff
  03C50 CB25              SLA     L                ;HL = HL * 2
  03C52 CB14              RL      H
  03C54 ED42              SBC     HL,BC            ;HL = smaller difference * 2 -
                                                   ;bigger difference
  03C56 CB21              SLA     C                ;BC = BC * 2
  03C58 CB10              RL      B
  03C5A CB23              SLA     E                ;E = smaller difference * 2
  03C5C CB12              RL      D                ;D = bigger difference * 2
  03C5E 7C                LD      A,H              ;A = H
  03C5F D9                EXX                      ;Switch register sets
  03C60 E1                POP     HL               ;Restore X2,Y2
  03C61 C1                POP     BC               ;Restore angle
  03C62 D1                POP     DE               ;Restore X1,Y1

; Calculate and plot next point

  03C63 CB27              SLA     A                ;A,7 into C-flag
                                                   ;C-flag = 1 ?
  03C65 D4A03C            CALL    NC,3CA0H         ;No: change X2 and Y2
                                                   ;according to the angle of
                                                   ;the smaller difference
  03C68 CDAF3C            CALL    3CAFH            ;Change X2 and Y2 according to
                                                   ;the angle of the bigger diff.
  03C6B CDC63C            CALL    3CC6H            ;PLOT  H , L
  03C6E 7A                LD      A,D              ;A = X1
  03C6F BC                CP      H                ;X1 = X2 ?
  03C70 2003              JR      NZ,3C75H         ;No: set next point

  03C72 7B                LD      A,E              ;A = Y1
  03C73 BD                CP      L                ;Y1 = Y2 ?
  03C74 C8                RET     Z                ;Yes: done.

  03C75 D9                EXX                      ;Switch register sets
  03C76 19                ADD     HL,DE            ;HL' = HL' + DE'
  03C77 7C                LD      A,H              ;A = H'
  03C78 D9                EXX                      ;Switch register sets
  03C79 18E8              JR      3C63H            ;Set next point

; 'TO' indicated ?

  03C7B 7E                LD      A,(HL)           ;A = next character
  03C7C FEBD              CP      0BDH             ;'TO' token ?
  03C7E 200D              JR      NZ,3C8DH         ;No: C-flag = 1

  03C80 23                INC     HL               ;PTP + 1
  03C81 AF                XOR     A                ;C-flag = 0
  03C82 C9                RET
```

```
; FCLS statement
; --------------

  03C83 2B               DEC     HL              ;Adjust PTP
  03C84 D7               RST     10H             ;Get next non-space character
  03C85 3E00             LD      A,00H           ;A = 00H
  03C87 C34A38           JP      384AH           ;Continue ar 384AH

  03C8A 00               NOP                     ;--

; A = -A, C-flag = 1

  03C8B ED44             NEG                     ;A = -A
  03C8D 37               SCF                     ;C-flag = 1
  03C8E C9               RET

; Swap upper and lower nibble of C, E = X-diff, D = Y-diff (see 3C40H)

  03C8F CD973C           CALL    3C97H           ;Swap nibbles of C
  03C92 5F               LD      E,A             ;E = X-diff
  03C93 78               LD      A,B             ;A = Y-diff
  03C94 57               LD      D,A             ;D = A
  03C95 18AE             JR      3C45H           ;Back to 3C45H

; Swap upper and lower nibble of C

  03C97 CB09             RRC     C               ;Rotate C left 4 times
  03C99 CB09             RRC     C
  03C9B CB09             RRC     C
  03C9D CB09             RRC     C
  03C9F C9               RET

; Change X2 and Y2 according to the angle of the smaller difference

  03CA0 CD973C           CALL    3C97H           ;Swap nibbles of C
  03CA3 CDAF3C           CALL    3CAFH           ;Change X2 and Y2
  03CA6 CD973C           CALL    3C97H           ;Swap nibbles of C
  03CA9 D9               EXX                     ;Switch register sets
  03CAA B7               OR      A               ;C-flag = 0
  03CAB ED42             SBC     HL,BC           ;HL' = HL' - BC'
  03CAD D9               EXX                     ;Switch register sets
  03CAE C9               RET

; Change X2 and Y2 according to the angle of the bigger difference

  03CAF CB79             BIT     7,C             ;Change Y2 ?
  03CB1 CABD3C           JP      Z,3CBDH         ;Yes: continue at 3CBDH

  03CB4 CB71             BIT     6,C             ;Increase X2 ?
  03CB6 C2BB3C           JP      NZ,3CBBH        ;No: continue at 3CBBH

  03CB9 24               INC     H               ;X2 + 1
  03CBA C9               RET

  03CBB 25               DEC     H               ;X2 - 1
  03CBC C9               RET

  03CBD CB71             BIT     6,C             ;Increment Y2 ?
  03CBF C2C43C           JP      NZ,3CC4H        ;No: continue at 3CC4H

  03CC2 2C               INC     L               ;Y2 + 1
  03CC3 C9               RET

  03CC4 2D               DEC     L               ;Y2 - 1
  03CC5 C9               RET


; SUB for PLOT
; PLOT  H , L  (AF)
;
; I: H = X-coordinate
```

```
;    L = Y-coordinate

  03CC6 C5              PUSH    BC              ;Save registers
  03CC7 D5              PUSH    DE
  03CC8 E5              PUSH    HL
  03CC9 7C              LD      A,H             ;Swap H and L
  03CCA 65              LD      H,L
  03CCB 6F              LD      L,A
  03CCC CD8A3B          CALL    3B8AH           ;PLOT  H , L
  03CCF E1              POP     HL              ;Restore registers
  03CD0 D1              POP     DE
  03CD1 C1              POP     BC
  03CD2 C9              RET


; XSHAPE statement
; ----------------
  03CD3 110303          LD      DE,0303H        ;Set bit mask for XSHAPE
  03CD6 1808            JR      3CE0H           ;Continue at 3CE0H


; NSHAPE statement
; ----------------

  03CD8 110000          LD      DE,0000H        ;Set bit mask for NSHAPE
  03CDB 1803            JR      3CE0H           ;Continue at 3CE0H


; SHAPE statement
; ---------------

  03CDD 110003          LD      DE,0300H        ;Set bit mask for SHAPE

  03CE0 ED531743        LD      (4317H),DE      ;Save bit mask
  03CE4 2B              DEC     HL              ;Adjust PTP
  03CE5 D7              RST     10H             ;Get next non-space character
  03CE6 CD1C2B          CALL    2B1CH           ;Get X coordinate
  03CE9 F5              PUSH    AF              ;Save X coordinate
  03CEA CF              RST     08H             ;Next character must be
  03CEB 2C              DEFB    ','             ;a comma
  03CEC CD1C2B          CALL    2B1CH           ;Get Y coordinate
  03CEF D1              POP     DE              ;D = X coordinate
  03CF0 5F              LD      E,A             ;E = Y coordinate
  03CF1 E5              PUSH    HL              ;Save PTP
  03CF2 C3A235          JP      35A2H           ;Continue at 35A2H

  03CF5 46              LD      B,(HL)          ;B = length of SHAPE table
  03CF6 23              INC     HL              ;Update SHAPE table pointer
  03CF7 3A1443          LD      A,(4314H)       ;A = SCALE factor
  03CFA B7              OR      A               ;SCALE = 0 ?
  03CFB 284B            JR      Z,3D48H         ;Yes: done.

  03CFD 4F              LD      C,A             ;C = SCALE factor
  03CFE 7E              LD      A,(HL)          ;A = entry from SHAPE table
  03CFF CB2F            SRA     A               ;Take upper nibble
  03D01 CB2F            SRA     A
  03D03 CB2F            SRA     A
  03D05 CB2F            SRA     A

  03D07 08              EX      AF,AF'
  03D08 3E01            LD      A,01H           ;A' = 1 = flag for processing
                                                ;of first segment (every table
                                                ;entry contains 2 segments)
  03D0A 08              EX      AF,AF'

  03D0B F5              PUSH    AF              ;Save table value
  03D0C CB2F            SRA     A               ;Direction: vertical ? (bit0=1)
  03D0E 383D            JR      C,3D4DH         ;Yes: continue at 3D4DH

  03D10 CB2F            SRA     A               ;Direction: left ? (bit1=1)
  03D12 3836            JR      C,3D4AH         ;Yes: continue at 3D4AH
```

```
  03D14 14              INC     D               ;X + 1 (direction: right)
  03D15 C5              PUSH    BC              ;Save registers
  03D16 D5              PUSH    DE
  03D17 E5              PUSH    HL
  03D18 6F              LD      L,A             ;L = table value
  03D19 3A1343          LD      A,(4313H)       ;A = FCOLOUR value
  03D1C F5              PUSH    AF              ;Save it
  03D1D 7D              LD      A,L             ;Table value bacl to A
  03D1E 2A1743          LD      HL,(4317H)      ;Take bit mask for operation
  03D21 A4              AND     H               ;Mask colour value
  03D22 AD              XOR     L
  03D23 321343          LD      (4313H),A       ;Store as new FCOLOUR value
  03D26 6A              LD      L,D             ;Coordinates in HL
  03D27 63              LD      H,E
  03D28 CD8A3B          CALL    3B8AH           ;PLOT  L , H
  03D2B F1              POP     AF              ;Restore original FCOLOUR value
  03D2C 321343          LD      (4313H),A       ;Store it in system RAM
  03D2F E1              POP     HL              ;Restore registers
  03D30 D1              POP     DE
  03D31 C1              POP     BC
  03D32 F1              POP     AF              ;Restore table value
  03D33 0D              DEC     C               ;SCALE factor > 1 ?
  03D34 20D5            JR      NZ,3D0BH        ;Yes: process same table value
                                                ;again (C times)
  03D36 C5              PUSH    BC              ;Save table counter
  03D37 08              EX      AF,AF'
  03D38 3D              DEC     A               ;Flag = 1 ?
  03D39 47              LD      B,A             ;B = flag - 1
  03D3A 08              EX      AF,AF'
  03D3B 04              INC     B               ;B = 0 ?
  03D3C 05              DEC     B               ;(then flag was 1)
  03D3D C1              POP     BC              ;Restore table counter
  03D3E 3A1443          LD      A,(4314H)       ;A = SCALE factor
  03D41 4F              LD      C,A             ;C = SCALE factor
  03D42 7E              LD      A,(HL)          ;A = table value
                                                ;Flag was 1 ?
  03D43 28C6            JR      Z,3D0BH         ;Yes: process second segment

  03D45 23              INC     HL              ;Table pointer + 1
  03D46 10AF            DJNZ    3CF7H           ;Next entry in SHAPE table

  03D48 E1              POP     HL              ;Restore PTP
  03D49 C9              RET

; Direction: left

  03D4A 15              DEC     D               ;X-coordinate - 1
  03D4B 18C8            JR      3D15H           ;Back to SCALE routine

; Direction: vertical

  03D4D CB2F            SRA     A               ;Direction: up ? (Bit1=1)
  03D4F 3803            JR      C,3D54H         ;Yes: continue at 3D45H

  03D51 1C              INC     E               ;Y-coordinate + 1 (down)
  03D52 18C1            JR      3D15H           ;Back to SCALE routine

; Direction: up

  03D54 1D              DEC     E               ;Y-coordinate - 1
  03D55 18BE            JR      3D15H           ;Back to SCALE routine


; SUB for tokenizing (see 392AH)
; Store Colour-Token into coded line.

  03D57 0C              INC     C               ;Character counter + 1
  03D58 23              INC     HL              ;Buffer pointer + 1
  03D59 EB              EX      DE,HL           ;DE = buffer pointer (on coded
                                                ;text)
```

```
                                                  ;HL = text pointer (on uncoded
                                                  ;text)
  03D5A 23                INC     HL              ;Text pointer + 1
  03D5B 12                LD      (DE),A          ;Store token in buffer
  03D5C 13                INC     DE              ;Buffer pointer + 1
  03D5D 0C                INC     C               ;Character counter + 1
  03D5E C3CC1B            JP      1BCCH           ;Continue at 1BCCH


; PLAY statement
; --------------

  03D61 CF                RST     08H             ;Next character must be
  03D62 28                DEFB    '('             ;a '('
  03D63 CDC23F            CALL    3FC2H           ;Get channel number - 1
  03D66 FE03              CP      03H             ;In range (1 to 3) ?
  03D68 D24A1E            JP      NC,1E4AH        ;No: ?FC Error

  03D6B F5                PUSH    AF              ;Save channel number
  03D6C CF                RST     08H             ;Next character must be
  03D6D 2C                DEFB    ','             ;a comma.
  03D6E CDC43F            CALL    3FC4H           ;Get octave number - 1
  03D71 FE08              CP      08H             ;In range (1 to 8) ?
  03D73 30F3              JR      NC,3D68H        ;No: ?FC Error

  03D75 3C                INC     A               ;Adjust octave number
  03D76 F5                PUSH    AF              ;Save octave number
  03D77 CF                RST     08H             ;Next character must be
  03D78 2C                DEFB    ','             ;a comma
  03D79 CD1C2B            CALL    2B1CH           ;Get note number
  03D7C B7                OR      A               ;Note number zero ?
  03D7D 283F              JR      Z,3DBEH         ;Yes: continue at 3DBEH

  03D7F FE1E              CP      1EH             ;In range (1 to 29)
  03D81 30E5              JR      NC,3D68H        ;No: ?FC Error

  03D83 CB27              SLA     A               ;A * 2 to get offset
  03D85 5F                LD      E,A             ;DE = note table offset
  03D86 1600              LD      D,00H
  03D88 CF                RST     08H             ;Next character must be
  03D89 2C                DEFB    ','             ;a comma
  03D8A E5                PUSH    HL              ;Save PTP

  03D8B 21CF3D            LD      HL,3DCFH        ;HL -> note table
  03D8E 19                ADD     HL,DE           ;Add offset
  03D8F 5E                LD      E,(HL)          ;DE = note frequency
  03D90 23                INC     HL
  03D91 56                LD      D,(HL)
  03D92 E1                POP     HL              ;Restore PTP
  03D93 C1                POP     BC              ;B = octave number
  03D94 05                DEC     B               ;Adjust to use as counter
                                                  ;Counter zero ?
  03D95 2806              JR      Z,3D9DH         ;Yes: continue at 3D9DH

  03D97 CB3A              SRL     D               ;Do DE = DE / 2 for every
  03D99 CB1B              RR      E               ;octave: double frequency
  03D9B 10FA              DJNZ    3D97H           ;Loop

  03D9D F1                POP     AF              ;A = channel number
  03D9E F5                PUSH    AF              ;Save it again
  03D9F CB27              SLA     A               ;Calculate register address
  03DA1 3C                INC     A               ;A = A * 2 + 1
  03DA2 CD2A3E            CALL    3E2AH           ;Write note freq. into PSG
  03DA5 CD1C2B            CALL    2B1CH           ;Get volume value
  03DA8 FE11              CP      11H             ;In range (0 to 16) ?
  03DAA 30BC              JR      NC,3D68H        ;No: ?FC Error

  03DAC D5                PUSH    DE              ;Save frequency setting
  03DAD 1E38              LD      E,38H           ;Select:
  03DAF 3E07              LD      A,07H           ;A = PSG register 7
  03DB1 CD323E            CALL    3E32H           ;Write to PSG
```

```
  03DB4 D1                POP     DE               ;Restore frequency setting
  03DB5 F1                POP     AF               ;Restore channel number

  03DB6 C608              ADD     A,08H            ;Calculate register address
  03DB8 CD323E            CALL    3E32H            ;Write to PSG
  03DBB CF                RST     08H              ;Next character must be
  03DBC 29                DEFB    ')'              ;a ')'
  03DBD C9                RET

  03DBE CF                RST     08H              ;Next character must be
  03DBF 2C                DEFB    ','              ;a comma
  03DC0 CD1C2B            CALL    2B1CH            ;Get amplitude value
  03DC3 FE11              CP      11H              ;In range (0 to 16) ?
  03DC5 D24A1E            JP      NC,1E4AH         ;No: ?FC Error

  03DC8 F1                POP     AF               ;Correct stack
  03DC9 F1                POP     AF               ;Get channel number
  03DCA 1E00              LD      E,00H            ;Amplitude = 0
  03DCC C3B63D            JP      3DB6H            ;Continue at 3DB6H


; Note table containing the frequency register values for the PSG

  03DCF 0000              DEFW    0000H
  03DD1 5D0D              DEFW    0D5DH
  03DD3 E70B              DEFW    0BE7H
  03DD5 9B0A              DEFW    0A9BH
  03DD7 020A              DEFW    0A02H
  03DD9 EB08              DEFW    08EBH
  03DDB F207              DEFW    07F2H
  03DDD 1407              DEFW    0714H
  03DDF 9C0C              DEFW    0C9CH
  03DE1 3C0B              DEFW    0B3CH
  03DE3 7309              DEFW    0973H
  03DE5 6B08              DEFW    086BH
  03DE7 8007              DEFW    0780H
  03DE9 5D0D              DEFW    0D5DH
  03DEB 5D0D              DEFW    0D5DH
  03DED 5D0D              DEFW    0D5DH
  03DEF 4A09              DEFW    094AH
  03DF1 9010              DEFW    1090H
  03DF3 C00E              DEFW    0EC0H
  03DF5 240D              DEFW    0D24H
  03DF7 680C              DEFW    0C68H
  03DF9 0C0B              DEFW    0B0CH
  03DFB D809              DEFW    09D8H
  03DFD C808              DEFW    08C8H
  03DFF A00F              DEFW    0FA0H
  03E01 EB0D              DEFW    0DEBH
  03E03 B40B              DEFW    0BB4H
  03E05 700A              DEFW    0A70H
  03E07 4A09              DEFW    094AH
  03E09 4808              DEFW    0848H

  03E0B F6B7              OR      0B7H             ;--
  03E0D C9                RET


; SUB for ? (not used)
; Write DE in X. If DE > 0 then VT = INT else VT = SNG

  03E0E CB7A              BIT     7,D              ;D negative ?
  03E10 2813              JR      Z,3E25H          ;No: Write DE in X as INT

  03E12 CDEF0A            CALL    0AEFH            ;VT = SNG
  03E15 7A                LD      A,D              ;A <- D <- E <- 00H
  03E16 53                LD      D,E              ;(Shift DE 8 bits to the left,
  03E17 1E00              LD      E,00H            ;overflow in A)
  03E19 B7                OR      A                ;C-flag = 0
  03E1A 1F                RRA                      ;ADE back 1 bit to the right
  03E1B CB1A              RR      D
```

```
  03E1D CB1B             RR      E
  03E1F 0691             LD      B,91H           ;B = Exp for 2 ^ 17
  03E21 CD6909           CALL    0969H           ;SFLOAT (X)
  03E24 C9               RET

  03E25 EB               EX      DE,HL           ;HL = DE
  03E26 CD9A0A           CALL    0A9AH           ;Write HL to X as INT
  03E29 C9               RET


; SUB for PSG programming
; Wwrites the 16 bit value in DE into PSG registers addressed by A
;
; I: A = register number of MSB data register
;    DE = register data

  03E2A 47               LD      B,A             ;Save register number in B
  03E2B D3F8             OUT     (0F8H),A        ;Select PSG register
  03E2D 7A               LD      A,D             ;Register data from D to A
  03E2E D3F9             OUT     (0F9H),A        ;Write data in PSG register
  03E30 05               DEC     B               ;register number - 1
  03E31 78               LD      A,B             ;Put it in A

; Now write LSB register with value in E

  03E32 D3F8             OUT     (0F8H),A        ;Select PSG register
  03E34 7B               LD      A,E             ;Register data from E to A
  03E35 D3F9             OUT     (0F9H),A        ;Write data in PSG register
  03E37 C9               RET


; PAINT statement
; ---------------

  03E38 AF               XOR     A               ;Number of parameters = 0
  03E39 2B               DEC     HL              ;Adjust PTP

  03E3A 3C               INC     A               ;Number of parameters + 1
  03E3B 08               EX      AF,AF'          ;A' = counter
  03E3C D7               RST     10H             ;Get next non-space character
  03E3D CD1C2B           CALL    2B1CH           ;Get next parameter
  03E40 F5               PUSH    AF              ;Save parameter on stack
  03E41 2B               DEC     HL              ;Adjust PTP
  03E42 D7               RST     10H             ;Get next non-space character
                                                 ;End reached ?
  03E43 280C             JR      Z,3E51H         ;Yes: continue at 3E51H

  03E45 FE2C             CP      2CH             ;Comma ?
  03E47 2005             JR      NZ,3E4EH        ;No: ?SN Error

  03E49 08               EX      AF,AF'          ;A = counter
  03E4A FE05             CP      05H             ;Maximum number of parameters
                                                 ;(4) exceeded ?
  03E4C 38EC             JR      C,3E3AH         ;No: continue at 3E3AH

  03E4E C39719           JP      1997H           ;?SN Error

; Evaluate the parameters on stack

  03E51 08               EX      AF,AF'          ;A = counter
  03E52 FE03             CP      03H             ;At least 2 parameters ?
  03E54 38F8             JR      C,3E4EH         ;No: ?SN error

  03E56 3D               DEC     A               ;Correct A to get the number
  03E57 3D               DEC     A               ;border colours
  03E58 321D43           LD      (431DH),A       ;Store this in A
  03E5B 111843           LD      DE,4318H        ;DE -> system RAM area
                                                 ;for PAINT statement
  03E5E 47               LD      B,A             ;Number of border colours
  03E5F 83               ADD     A,E             ;DE -> last byte in
  03E60 5F               LD      E,A             ;system RAM area for PAINT
```

```
  03E61 F1              POP     AF              ;Get value from stack
  03E62 3D              DEC     A               ;-1
  03E63 FE04            CP      04H             ;Colour value in range ?
  03E65 D24A1E          JP      NC,1E4AH        ;No: ?FC Error

  03E68 12              LD      (DE),A          ;Store in system RAM
  03E69 1B              DEC     DE              ;pointer - 1
  03E6A 10F5            DJNZ    3E61H           ;Next parameter

  03E6C F1              POP     AF              ;A = Y coordinate
  03E6D FE66            CP      66H             ;Y coordinate in range ?
  03E6F 30F4            JR      NC,3E65H        ;No: ?FC Error

  03E71 57              LD      D,A             ;D = Y coordinate
  03E72 F1              POP     AF              ;A = X coordinate
  03E73 FEA0            CP      0A0H            ;X coordinate in range ?
  03E75 30EE            JR      NC,3E65H        ;No: ?FC Error

  03E77 5F              LD      E,A             ;E = X coordinate
  03E78 E5              PUSH    HL              ;Save PTP
  03E79 21FFFF          LD      HL,0FFFFH       ;HL = -1 (flag for routine end)
  03E7C E5              PUSH    HL              ;Save flag
  03E7D EB              EX      DE,HL           ;HL = Y,X

; Set next point

  03E7E 221E43          LD      (431EH),HL      ;Save coordinates in system RAM
  03E81 CD043F          CALL    3F04H           ;Set FCOLOUR if border not
                                                ;reached
  03E84 2806            JR      Z,3E8CH         ;Continue at 3E8CH when border
                                                ;has been reached
  03E86 2D              DEC     L               ;X - 1
  03E87 3EFF            LD      A,0FFH          ;A = -1
  03E89 BD              CP      L               ;X = -1 ?
  03E8A 20F5            JR      NZ,3E81H        ;No: next point

  03E8C 2C              INC     L               ;X + 1
  03E8D 7D              LD      A,L             ;A = X-coordinate
  03E8E 322043          LD      (4320H),A       ;Save it (left edge)
  03E91 2A1E43          LD      HL,(431EH)      ;Last coordinates to HL
  03E94 2C              INC     L               ;X + 1
  03E95 3EA0            LD      A,0A0H          ;A = 160
  03E97 BD              CP      L               ;X = 160 ?
  03E98 2805            JR      Z,3E9FH         ;Yes: continue at 3E9FH

  03E9A CD043F          CALL    3F04H           ;Set FCOLOUR if border not
                                                ;reached
  03E9D 20F5            JR      NZ,3E94H        ;If border not reached:
                                                ;next point
  03E9F 2D              DEC     L               ;X - 1
  03EA0 7D              LD      A,L             ;A = X-coordinate
  03EA1 322143          LD      (4321H),A       ;Save it (right edge)
  03EA4 3A1F43          LD      A,(431FH)       ;A = Y-coordinate
  03EA7 A7              AND     A               ;Y = 0 ?
  03EA8 280C            JR      Z,3EB6H         ;Yes: continue at 3EB6H

  03EAA 3D              DEC     A               ;Y - 1
  03EAB 67              LD      H,A             ;H = Y-coordinate
  03EAC CDC33E          CALL    3EC3H           ;Check Y-coordinate
  03EAF 3A1F43          LD      A,(431FH)       ;A = Y-coordinate
  03EB2 FE65            CP      65H             ;Y = 102 ?
  03EB4 2805            JR      Z,3EBBH         ;Yes: continue at 3EBBH

  03EB6 3C              INC     A               ;Y + 1
  03EB7 67              LD      H,A             ;H = Y-coordinate
  03EB8 CDC33E          CALL    3EC3H           ;Check Y-coordinate
  03EBB 3EFF            LD      A,0FFH          ;A = -1
  03EBD E1              POP     HL              ;Restore Y-coordinate
  03EBE BC              CP      H               ;Flag reached ?
  03EBF 20BD            JR      NZ,3E7EH        ;No: next point
```

```
  03EC1 E1              POP     HL              ;Restore PTP
  03EC2 C9              RET


; Check Y-coordinate
;
; I: H = Y-coordinate

  03EC3 0EFF            LD      C,0FFH          ;C = 255 (flag)
  03EC5 3A2043          LD      A,(4320H)       ;A = X-coordinate of left edge
  03EC8 6F              LD      L,A             ;L = left edge
  03EC9 3A2143          LD      A,(4321H)       ;A = X-coordinate right edge
  03ECC 95              SUB     L               ;A = distance between left
                                                ;and right edge
  03ECD 47              LD      B,A             ;B = distance
  03ECE 04              INC     B               ;B + 1
  03ECF C8              RET     Z               ;Done when both edges are
                                                ;identical

; Check complete line of new Y-coordinate

  03ED0 C5              PUSH    BC              ;Save register
  03ED1 E5              PUSH    HL
  03ED2 CDF13E          CALL    3EF1H           ;Border colour at L,H reached ?
  03ED5 E1              POP     HL              ;Restore registers
  03ED6 C1              POP     BC
  03ED7 2006            JR      NZ,3EDFH        ;No: continue at 3EDFH

  03ED9 0EFF            LD      C,0FFH          ;Yes: flag = 255
  03EDB 2C              INC     L               ;X + 1
  03EDC 10F2            DJNZ    3ED0H           ;Check next X-coordinate

  03EDE C9              RET


; Border colour not reaches at new Y-coordinate

  03EDF AF              XOR     A               ;A = 0
  03EE0 B9              CP      C               ;Flag = 0 ?
                                                ;(This area already recognized)
  03EE1 28F8            JR      Z,3EDBH         ;Yes: continue at 3EDBH

  03EE3 0E02            LD      C,02H           ;C = 2 (for 1963H)
  03EE5 C5              PUSH    BC              ;Save BC

  03EE6 CD6319          CALL    1963H           ;Stack has still room enough ?
  03EE9 C1              POP     BC              ;Restore BC
  03EEA D1              POP     DE              ;RET address to DE
  03EEB E5              PUSH    HL              ;Save X,Y coordinates
  03EEC D5              PUSH    DE              ;Put RET address back on stack
  03EED 0E00            LD      C,00H           ;Flag = 0
  03EEF 18EA            JR      3EDBH           ;Continue at 3EDBH


; Border colour reached at point L,H ?
; Z-flag = 0: No
; Z-flag = 1: Yes

  03EF1 CD3A3F          CALL    3F3AH           ;A = CPOINT( L,H )
  03EF4 E5              PUSH    HL              ;Save coordinated
  03EF5 211D43          LD      HL,431DH        ;HL -> number of border colours
  03EF8 46              LD      B,(HL)          ;B = number of border colours
  03EF9 211943          LD      HL,4319H        ;HL -> border colour table
  03EFC BE              CP      (HL)            ;Border colour reached ?
  03EFD 2803            JR      Z,3F02H         ;Yes: done.

  03EFF 23              INC     HL              ;Pointer + 1
  03F00 10FA            DJNZ    3EFCH           ;Check next colour

  03F02 E1              POP     HL              ;Restore coordinates
```

```
  03F03 C9                RET


; Set new colour for point L,H in case border colour is not reached yet.

  03F04 E5                PUSH    HL                ;Save coordinates
  03F05 CDF13E            CALL    3EF1H             ;Border colour reached ?
  03F08 F5                PUSH    AF                ;Save flags
  03F09 41                LD      B,C               ;B = X MOD 4 (from CPOINT)
  03F0A 3A1943            LD      A,(4319H)         ;A = new FCOLOUR
  03F0D 4F                LD      C,A               ;C = A
  03F0E C4B03B            CALL    NZ,3BB0H          ;If border colour not reached:
                                                    ;set new colour
  03F11 F1                POP     AF                ;Restore flags
  03F12 E1                POP     HL                ;Restore coordinates
  03F13 C9                RET


; Old KEYPAD1 routine (no longer used)

  03F14 16FE              LD      D,0FEH
  03F16 1802              JR      3F1AH

; Old KEYPAD2 routine (no longer used)

  03F18 16F7              LD      D,0F7H
  03F1A CD873A            CALL    3A87H
  03F1D 6F                LD      L,A
  03F1E 2600              LD      H,00H
  03F20 C9                RET


; SUB for processing expressions (at 2337H)
; Continuation of 3F7AH: interception of Colour-BASIC functions

  03F21 FE91              CP      91H               ;Token for SOUND ?
  03F23 CAF436            JP      Z,36F4H           ;Yes: continue at 36F4H

  03F26 FE8A              CP      8AH               ;Token for SCALE ?
  03F28 CA0A37            JP      Z,370AH           ;Yes: continue at 370AH

  03F2B FE80              CP      80H               ;Token for COLOUR ?
  03F2D CA6136            JP      Z,3661H           ;Yes: continue at 3661H

  03F30 C30525            JP      2505H             ;Continue at 2505H


; VERIFY statement
; ----------------

  03F33 23                INC     HL                ;PTP + 1
  03F34 C3292C            JP      2C29H             ;Continue at 2C29H

  03F37 6E                LD      L,(HL)            ;--
  03F38 63                LD      H,E
  03F39 77                LD      (HL),A


; SUB for CPOINT and PAINT
; This routine takes the colour value for pixel at (X,Y)
;
; I: L = X coordinate
;    H = Y coordinate
; O: A = colour value (1 to 4) - 1

  03F3A 7D                LD      A,L               ;A = X coordinate
  03F3B 6C                LD      L,H               ;HL = Y coordinate
  03F3C 2600              LD      H,00H
  03F3E 54                LD      D,H               ;DE = Y coordinate
  03F3F 5D                LD      E,L
  03F40 29                ADD     HL,HL             ;Multiply Y coordinate by 40
```

```
  03F41 29              ADD     HL,HL           ;(40 bytes/ line)
  03F42 19              ADD     HL,DE           ;Result in HL
  03F43 29              ADD     HL,HL           ;HL = offset from start of FGR
  03F44 29              ADD     HL,HL           ;page to start of line (Y)
  03F45 29              ADD     HL,HL
  03F46 5F              LD      E,A             ;Multiply X coordinate by 4
  03F47 CB3B            SRL     E               ;(4 pixels/byte)
  03F49 CB3B            SRL     E               ;E = offset from start of line
                                                ;to X
  03F4B 1648            LD      D,48H           ;DE = start of FGR page + E
  03F4D 19              ADD     HL,DE           ;HL -> FGR memory location
                                                ;that contains (X,Y)
  03F4E E603            AND     03H             ;Now make counter for
  03F50 3C              INC     A               ;getting the proper 2 bits for
  03F51 4F              LD      C,A             ;X coordinate within the byte
  03F52 47              LD      B,A             ;Result in B as counter
  03F53 7E              LD      A,(HL)          ;Load byte from FGR memory

  03F54 07              RLCA                    ;Rotate until the proper 2
  03F55 07              RLCA                    ;bits are in bit 1 and 0
  03F56 10FC            DJNZ    3F54H

  03F58 E603            AND     03H             ;Clear bits 7 to 2
  03F5A C9              RET

  03F5B 00              NOP                     ;--


; X = CPOINT(x-coordinate,y-coordinate)

  03F5C CF              RST     08H             ;Next character must be
  03F5D 28CD            DEFB    ')'             ;a '('
  03F5E CD1C2B          CALL    2B1CH           ;Get X coordinate
  03F61 F5              PUSH    AF              ;and put it on stack
  03F62 CF              RST     08H             ;Next character must be
  03F63 2C              DEFB    ','             ;a comma
  03F64 CD1C2B          CALL    2B1CH           ;Get Y coordinate
  03F67 F5              PUSH    AF              ;and put it on stack
  03F68 CF              RST     08H             ;Next character must be
  03F69 29              DEFB    ')'             ;a ')'
  03F6A D1              POP     DE              ;D = Y coordinate
  03F6B F1              POP     AF              ;A = X coordinate
  03F6C EB              EX      DE,HL           ;H = Y coordinate, DE = PTP
  03F6D D5              PUSH    DE              ;Save PTP
  03F6E 6F              LD      L,A             ;L = X coordinate
  03F6F CD3A3F          CALL    3F3AH           ;Get FGR memory contents for
                                                ;specified coordinate

  03F72 6F              LD      L,A             ;HL = A
  03F73 2600            LD      H,00H
  03F75 CD9A0A          CALL    0A9AH           ;Write HL to X as INT
  03F78 E1              POP     HL              ;Restore PTP
  03F79 C9              RET


; SUB for processing of expressions
; Interception of Colour-BASIC functions

  03F7A CAFE27          JP      Z,27FEH         ;USR token found:
                                                ;Continue at 27FEH
  03F7D FEFF            CP      0FFH            ;Colour-BASIC token ?
  03F7F C20425          JP      NZ,2504H        ;No: continue at 2504H

  03F82 D7              RST     10H             ;Increase PTP, next token in A
  03F83 23              INC     HL              ;PTP + 1

  03F84 FE82            CP      82H             ;Token for KEYPAD ?
  03F86 CA0F3A          JP      Z,3A0FH         ;Yes: continue at 3A0FH

  03F89 FE8F            CP      8FH             ;Token for CPOINT ?
  03F8B 28CF            JR      Z,3F5CH         ;Yes: continue at 3F5CH
```

```
  03F8D FE83            CP      83H             ;Token for JOY ?
  03F8F CA1F3A          JP      Z,3A1FH         ;Yes: continue at 3A1FH

  03F92 C3213F          JP      3F21H           ;Continue at 3F21H


; SOUND statement
; ---------------

  03F95 CD1C2B          CALL    2B1CH           ;Get PSG register number
  03F98 FE10            CP      10H             ;In range (0 to 15) ?
  03F9A D24A1E          JP      NC,1E4AH        ;No: ?FC Error

  03F9D F5              PUSH    AF              ;Save PSG register number
  03F9E CF              RST     08H             ;Next character must be
  03F9F 2C              DEFB    ','             ;a comma
  03FA0 CD1C2B          CALL    2B1CH           ;Get register data
  03FA3 5F              LD      E,A             ;E = register data
  03FA4 F1              POP     AF              ;A = PSG register number
  03FA5 C3323E          JP      3E32H           :Write to PSG


; CHAR statement
; --------------

  03FA8 CDC23F          CALL    3FC2H           ;A = char value - 1
  03FAB FE04            CP      04H             ;Value in range (1 to 4) ?
  03FAD D24A1E          JP      NC,1E4AH        ;No: ?FC Error
  03FB0 E603            AND     03H             ;Mask bits
  03FB2 07              RLCA                    ;Rotate to proper position
  03FB3 07              RLCA
  03FB4 07              RLCA
  03FB5 47              LD      B,A             ;Put it in B
  03FB6 3A1C43          LD      A,(431CH)       ;A = port 255 output status
  03FB9 E6E7            AND     0E7H            ;Clear the old bits for CHAR
  03FBB B0              OR      B               ;Set the new bits
  03FBC D3FF            OUT     (0FFH),A        ;Activate new CHAR
  03FBE 321C43          LD      (431CH),A       ;and store status in system RAM
  03FC1 C9              RET


; Process expression at (HL) and return result - 1 in A
;
; I: HL = PTP on BASIC expression
; O: A = result - 1

  03FC2 2B              DEC     HL              ;PTP - 1
  03FC3 D7              RST     10H             ;Get next non-space character
  03FC4 CD1C2B          CALL    2B1CH           ;Get result
  03FC7 3D              DEC     A               ;A - 1
  03FC8 C9              RET


; Set character and cursor colour (not used)

  03FC9 CD7A30          CALL    307AH
  03FCC 23              INC     HL
  03FCD CD7A30          CALL    307AH
  03FD0 2B              DEC     HL
  03FD1 C9              RET


; SUB for LIST (see 2B91H)
; No token found. Output a text-constant ?

  03FD2 FE22            CP      22H             ;Text-constant ?
  03FD4 C2892B          JP      NZ,2B89H        ;No: return to LIST

  03FD7 03              INC     BC              ;Buffer pointer + 1
  03FD8 15              DEC     D               ;Counter - 1
```

```
  03FD9 C8              RET     Z               ;Return when line buffer full

  03FDA 7E              LD      A,(HL)          ;Get character
  03FDB FE22            CP      22H             ;End of text reached ?
  03FDD 23              INC     HL              ;Text pointer + 1
  03FDE 02              LD      (BC),A          ;Put character in buffer
  03FDF CA892B          JP      Z,2B89H         ;Yes: return to LIST

  03FE2 18F3            JR      3FD7H           ;Next character


; BGRD statement
; --------------

  03FE4 2B              DEC     HL              ;Adjust PTP
  03FE5 D7              RST     10H             ;Get next non-space character
  03FE6 0604            LD      B,04H           ;Set BGRD value to 4
                                                ;parameter present after BGRD ?
  03FE8 CABD38          JP      Z,38BDH         ;No: continue at 38BDH

  03FEB CDC23F          CALL    3FC2H           ;A = bgrd value - 1
  03FEE FE04            CP      04H             ;Value in range (1 to 4) ?
  03FF0 D24A1E          JP      NC,1E4AH        ;No: ?FC Error

  03FF3 E603            AND     03H             ;Mask bits
  03FF5 0F              RRCA                    ;Rotate to proper position
  03FF6 0F              RRCA
  03FF7 47              LD      B,A             ;Put result in B
  03FF8 3A1C43          LD      A,(431CH)       ;A = port 255 output status
  03FFB E63F            AND     3FH             ;Clear old bits for BGRD n
  03FFD C3C238          JP      38C2H           ;Continue at 38C2H

; End of BASIC interpreter
```